INITIATION AUX BASES DE DONNÉES POUR MICRO-ORDINATEURS

application à dBASE II pour :

AMSTRAD

CPC 6128 et PCW 8256

PAR

ROBERT A. BYERS



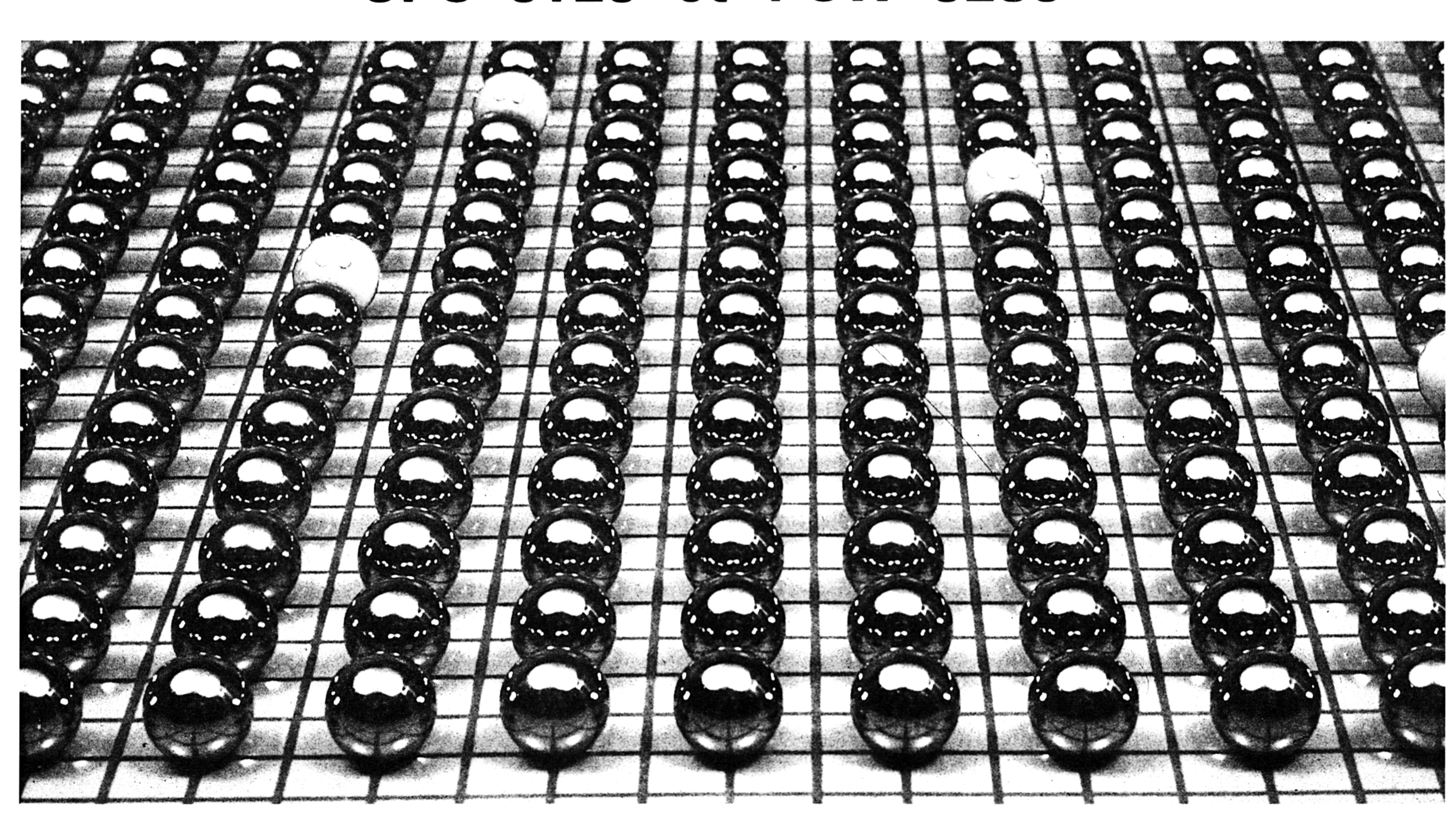


INITIATION AUX BASES DE DONNÉES POUR MICRO-ORDINATEURS

application à dBASE II pour :

AMSTRAD

CPC 6128 et PCW 8256



ROBERT A. BYERS

Traduction : La Commande Electronique.
Illustrations de Bernard GENIN
Composition et mise en page : Imprimerie Durand.
Imprimé en France par DM Impressions.
Éditeur : La Commande Électronique
7, rue des Prias
27920 Saint-Pierre-de-Bailleul

Ce livre a été publié en anglais sous le titre « Everyman's Database Primer featuring dBASE II » par Ashton Tate Publishing Group.

Dépôt légal, 4e trimestre 1985.
ISBN 2.905 350.06.7

REMERCIEMENTS

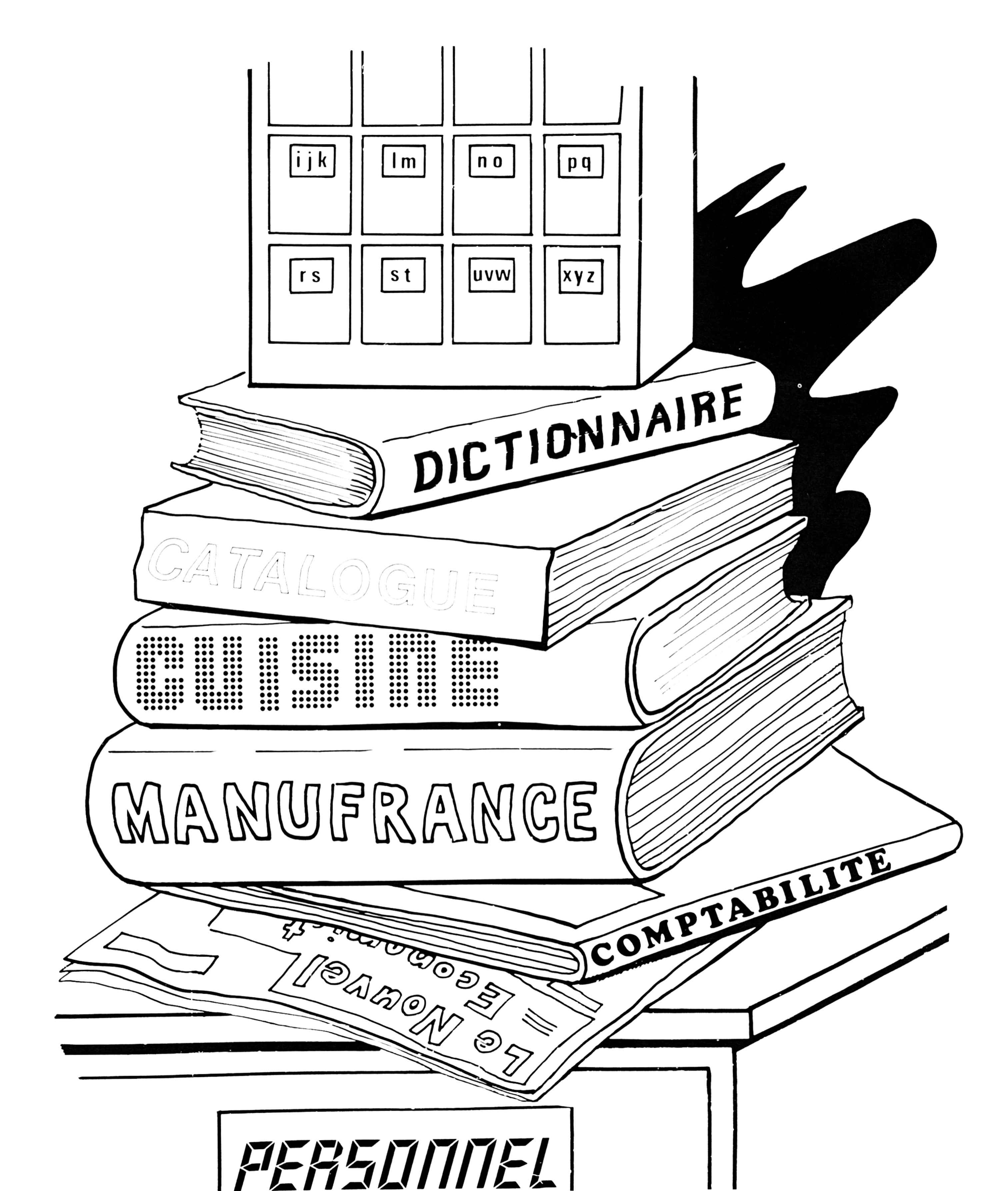
Je souhaite remercier tous ceux qui ont contribué d'une manière ou d'une autre à cet ouvrage. George Tate et Wayne Ratliff qui m'ont suggéré ce livre et qui ont dispensé leurs encouragements et leurs conseils avisés tout au long de cette entreprise. Virginia Bare, qui a corrigé patiemment et habilement le manuscrit. Mes fils, Kenneth et Robert Jr. qui ont relu de bon cœur le premier manuscrit et ont prodigué leur soutien et leurs encouragements. Et enfin, à Charlene Eber, responsable de mon intérêt pour les systèmes de gestion de base de données sur micro-ordinateur, la place d'honneur qui lui revient.

TABLE DES MATIÈRES

PREMIERE PARTIE

La première partie nous présente les bases de données et nous explique leur fonctionnement. Les bases de données sont très répandues autour de nous et nous ne soupçonnons pas vraiment la place qu'elles occupent dans notre vie quotidienne. Un « système de gestion de base de données » est simplement un moyen d'enregistrer et de manipuler ces bases de données que nous connaissons tous, à l'aide d'un ordinateur.

Nous examinerons de plus près des exemples de bases de données courantes, ce qui nous permettra de disséquer les parties qui la composent et d'en comprendre ainsi leur fonctionnement à partir duquel nous obtenons des informations organisées. Le processus est tellement simple que nous commencerons tout de suite. Nous allumerons notre ordinateur et nous créerons notre première base de données informatique.



ijk
lm
no
pq
rs
st
uvw
xyz
DICTIONNAIRE
CATALOGUE
CUISINE
MANUFRANCE
COMPTABILITE
Le Nouvel
PERSONNEL

CHAPITRE I

BASE DE DONNEES

Le mot « Base de données » fait partie du jargon informatique, il représente pourtant un élément essentiel dans notre vie de tous les jours. Une base de données est une collection d'informations organisées et présentées pour être utilisées à des fins particulières.

L'annuaire téléphonique est une des bases de données les plus connues. Cette base de données courante sous forme imprimée, contient les noms, adresses et numéros de téléphone des particuliers, des différentes professions et administrations. Les adresses et numéros de téléphone n'ont que peu de valeur par eux-mêmes. Ils sont surtout utiles lorsqu'ils sont reliés à un nom.

Lorsque nous réfléchissons à cela, le nombre de bases de données que nous connaissons est finalement surprenant. La plupart des bases de données les plus connues sont : un dictionnaire, un livre de recettes de cuisine, un catalogue de vente par correspondance, une encyclopédie, le catalogue d'une bibliothèque, votre chéquier, etc... Les autres bases de données sont plus ou moins connues, c'est, par exemple, un fichier du personnel, les cours de la bourse dans un journal, un grand livre de comptabilité, etc...

Maintenant, pourquoi considérons-nous ces exemples comme étant des bases de données ? Pourquoi un journal ou un roman n'est-il pas considéré comme une base de données ? Après tout, ils contiennent également de l'information. La raison est que dans chaque exemple énuméré ci-dessus, l'information est présentée de telle façon qu'il est facile à l'utilisateur de localiser l'élément particulier d'information qui l'intéresse. Si l'on prend l'annuaire par exemple, les numéros de téléphone et les adresses sont reliés au nom. Les noms sont présentés par ordre alphabétique, ainsi on peut les retrouver facilement. Vous retrouverez le nom et vous aurez le numéro de téléphone. Le nom est la clé d'utilisation de l'annuaire. L'exemple du dictionnaire est presque identique. Il se présente sous forme d'un mot et d'une définition. Les mots sont présentés par ordre alphabétique de façon à être retrouvés. La définition est reliée au mot. Le mot est la clé pour utiliser le dictionnaire.

L'élément commun à tous ces exemples est *l'information organisée, présentée de telle façon qu'il soit facile de la retrouver* — par l'utilisation d'une clé. En d'autres termes, l'information qui peut être présentée comme un tableau (lignes et colonnes) peut constituer une base de données. Quelques exemples de titres de colonnes dans des tableaux considérés comme des bases de données sont montrés à la Figure 1-1.

Exemples :	En-têtes de colonnes			
ANNUAIRE :	NOM	ADRESSE	No TELEPHONE	
DICTIONNAIRE :	MOT	DEFINITION		
CATALOGUE :	REFERENCE	DESIGNATION	POIDS	TAILLE
	PRIX	No PIECE		
ETAT DE STOCK :	QUANTITE	RAYON	MINI	MAXI

Figure 1-1. Exemples d'en-têtes de colonnes de bases de données "papier" courantes

Dès maintenant vous devez avoir une idée de ce qu'est une base de données et vous vous demandez : « Bon, mais qu'est-ce qu'une base de données sur ordinateur ? Que puis-je faire avec cette base que je ne pourrais pas faire si je ne l'ai pas ? ». Les bases de données sur ordinateur effectueront ce que vous pourrez faire vous-même. Cependant, certaines fonctions exécutables sans ordinateur peuvent ne pas être pratiques. Par exemple, nous avons tous rencontré un morceau de papier sur lequel est griffoné un numéro de téléphone — pas de nom, simplement un numéro. Si nous voulons trouver à qui correspond ce numéro, l'annuaire ne nous sera pas d'une aide très efficace. Si, cependant, la liste des téléphones est une base de données d'ordinateur, nous pouvons demander à l'ordinateur à qui appartient ce numéro et aussitôt le nom apparaîtra.

Prenons un autre exemple, supposons que vous souhaitez obtenir le numéro de téléphone d'une personne appelée Martin qui habite rue du Repos à Saint-Etienne.

Vous pouvez demandez à l'ordinateur de consulter son fichier annuaire de Saint-Etienne sur les noms, adresses et numéros de téléphone de tous les Martin de la rue du Repos. Il pourrait vous en sortir aucun, mais il en exclura certainement un grand nombre.

L'ordinateur n'est pas la panacée. Il ne peut pas faire ce que vous ne savez pas faire. Mais — ce mais est d'importance ! — il peut vous aider à faire les choses que vous souhaitez faire rapidement et facilement. C'est un outil pour vous aider à accomplir des tâches qui ne sont pas vraiment pratiques sans l'ordinateur.

Si l'on utilise un annuaire comme exemple, une simple base de données pourrait ressembler à la Figure 1-2.

NOM	ADRESSE	TELEPHONE
Boulanger Robert	13 Rue de la Terasse	565 78 00
Fouquet Sébastien	69 Rue de la Trésorerie	112 28 30
Lamotte Robert	4 Rue de la Laiterie	558 32 44
Legendre Jacques	125 Rue des Ormes	228 30 42
Lenoir Christian	11 Rue du Cimetière	771 21 39
Robertson Stanislas	506 Avenue Foch	333 43 12
Poupon Pierre	4 Boulevard du Temple	701 13 59
Schmitt Wilfrid	41 Rue de la Canebière	808 32 16

Figure 1-2. Un exemple de base de données

Bien entendu, une base de données réelle peut contenir beaucoup, beaucoup plus d'éléments d'information. En fait, la base de données ci-dessus est plus à sa place dans un petit carnet de notes que dans un ordinateur. Vous pouvez le transporter partout avec vous, vous en servir pour prendre des notes, ce qui revient moins cher ! Pour qu'un ordinateur devienne rentable, vous avez besoin de beaucoup d'informations — en général, de tellement d'informations que vous ne pouvez pas effectivement les manipuler sans l'ordinateur. Le but de cet ouvrage est de vous apprendre en quoi consiste une base de données et comment s'en servir. Par conséquent, nous allons utiliser de très petites bases de données comme exemples. Les mêmes principes s'appliquent exactement à cet annuaire personnel qu'aux pages

blanches de l'annuaire de la ville de Saint-Etienne. Il n'y a pas d'autres différences qu'un plus grand volume d'informations dans les pages blanches.

Pour mieux comprendre les bases de données sur micro-ordinateur, il faut savoir :

- comment les concevoir
- comment les fabriquer
- comment les utiliser et
- comment les modifier

Nous allons construire une base de données d'ordinateur à partir de notre exemple simple d'agenda téléphonique. Pour ce faire, nous allons utiliser le système de base de données de micro-informatique « dBASE II ». dBASE II représente le meilleur système de gestion de base de données disponible à l'heure actuelle sur micro-ordinateurs. Un système de gestion de base de données est un système qui peut être installé sur un ordinateur, s'occuper de tous les détails pratiques nécessaires à une base de données, et permettre à l'utilisateur d'utiliser, manipuler, et modifier les contenus de la base de données.

Si vous possédez ou avez accès à un ordinateur avec dBASE II, vous pouvez suivre les instructions de cet ouvrage et travailler en utilisant votre ordinateur. Si vous n'avez ni micro-ordinateur ni dBASE II, vous serez toutefois capable de suivre sans difficulté. Une description de ce que vous faites et de ce que fait l'ordinateur, sera fournie à chaque étape. Ce que « fait » l'ordinateur apparaîtra sur les illustrations d'écran en caractères normaux : ce que vous « faites », ce que vous *tapez*, vous apparaîtra en majuscules.

D'UN REPERTOIRE TELEPHONIQUE A UNE BASE DE DONNEES D'ORDINATEUR : UNE QUESTION DE TERMINOLOGIE

Examinons de nouveau les données de notre répertoire téléphonique personnel.

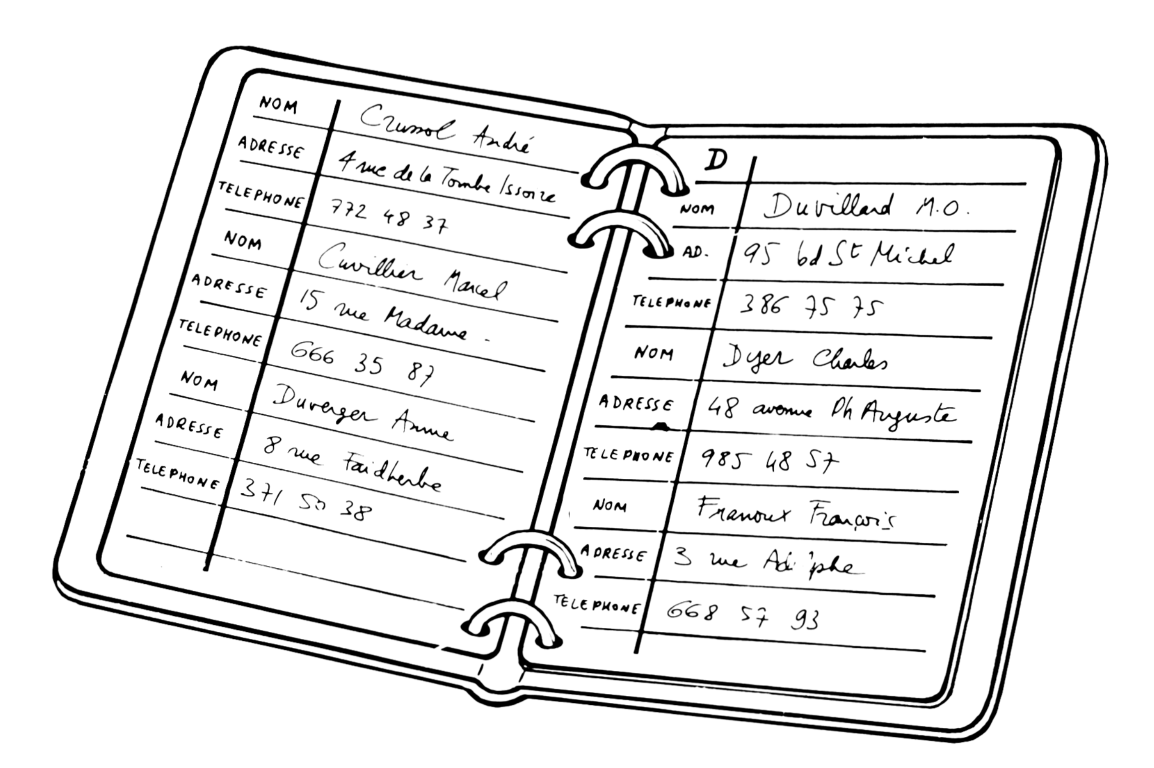
L'enregistrement

Lamotte Robert 4 Rue de la Laiterie 558 32 44

est appelé un ENREGISTREMENT. Les parties d'information qui constituent l'enregistrement sont montrées horizontalement affichées en lignes sur une page écran. Notre répertoire a huit lignes (huit ENREGISTREMENTS) huit jeux de nom + adresse + numéro de téléphone.

La rubrique

Si nous avions tracé des lignes entre les noms et les adresses et entre les adresses et les numéros de téléphone dans notre répertoire, nous pourrions isoler les colonnes d'information identiques. Nous pourrions séparer trois groupes de données disposées verticalement : une colonne de noms, une colonne d'adresses et une colonne de numéros de téléphone. En termes d'ordinateur, ces colonnes sont appelées des rubriques ou CHAMPS.



Donc, l'exemple ci-dessus a HUIT enregistrements et TROIS rubriques.

Noms de rubriques

Les titres de colonnes, NOM, ADRESSE, et NUMERO DE TELEPHONE, sont appelés des NOMS DE RUBRIQUES. C'est un seul terme — NOM DE RUBRIQUE.

Théoriquement, une base de données d'ordinateur est de même nature que ce que nous pourrions créer avec du papier et un crayon. Evidemment, une base de données papier et une base de données d'ordinateur existent pour être UTILISEES. Une question vient alors à l'esprit : Qu'est-ce que nous faisons actuellement avec notre répertoire téléphonique ?

Nous écrivons de nouveaux rendez-vous, nous changeons parfois les noms, les adresses ou les numéros de téléphone des gens qui se sont déplacés, qui ont un nouveau numéro de téléphone, qui sont mariés ou divorcés. Nous rayons quelquefois des personnes (ou nous les effaçons, si nous n'avions pas prévu de les conserver...). Nous pouvons utiliser l'information qui est stockée dans ce répertoire, lorsque nous voulons téléphoner, aller à une soirée, à une adresse particulière, envoyer une lettre, etc...

Votre répertoire téléphonique (si vous le maintenez à jour) nécessite un processus de modification et, à un moment donné, vous fournira l'information que vous attendez. C'est exactement la même chose avec la base de données de votre ordinateur.

Vous pouvez très facilement ajouter, retirer ou modifier l'information dans une base de données d'ordinateur. De plus, vous pouvez obtenir l'information de votre base de données pour la consulter, et faire tout cela très rapidement.

Dans nos activités quotidiennes, nous sommes toujours en train d'ajouter et de soustraire de l'information à la main, la modifier, extraire ce que nous voulons voir et ignorer ce que nous ne voulons pas voir. Cette activité est le processus de base de notre intelligence (ce processus est extrêmement courant).

Mais, nous sommes tellement habitués à avoir autour de nous des bases de données qu'il doit bien y avoir quelque chose qui gère cette information dans l'ordinateur, entre nous et lui, aussi nous allons en parler. Nous avons besoin de nous faire à l'idée que cette information se trouve à l'intérieur de l'ordinateur. Nous devons savoir que nous pouvons extraire l'information dont nous avons besoin pour l'utiliser. Après une certaine habitude, nous serons surpris de ce que peut faire actuellement une base de données d'ordinateur pour nos besoins en information.

Utiliser un ordinateur est aussi facile qu'utiliser un répertoire téléphonique. Comme tout ce que vous faites dans la vie, cela nécessite un peu de réflexion, un peu de préparation et un peu de pratique. Vous avez besoin de connaître comment créer, utiliser et modifier l'information qui est stockée. Dans le processus d'apprentissage, il ne sera pas nécessaire de réinventer la roue. Vous apprenez simplement une

nouvelle fonction (un nouveau jeu de mécanisme pour une nouvelle machine) construite pour assister vos efforts dans la recherche et le traitement de toutes sortes d'informations qui vous sont déjà familières.

Nous allons donc vous initier aux bases de données d'ordinateur. Ensemble nous allons construire et utiliser des bases simples. Afin que vous puissiez facilement faire la liaison entre les principes et les exemples de la pratique courante, nous utiliserons dBASE II. dBASE II est représentatif des systèmes de gestion de base de données couramment utilisés sur micro-ordinateurs.

QUELQUES SIMILITUDES

Pour illustrer le concept sous-tendu par un système de gestion de base de données, examinons un autre exemple simple. Imaginons que vous ayez besoin d'une pièce détachée pour votre voiture. Vous allez dans un garage et vous demandez à l'employé la pièce qu'il vous faut. Il recherche la pièce dans la liste d'un catalogue de pièces détachées.

- Le premier livre lui donne le numéro d'identification de la pièce.
- Il recherche alors ce numéro de pièce dans un autre livre. Ce livre le renseigne sur la position de la pièce détachée dans le magasin.
- Après avoir obtenu la pièce, il utilise de nouveau le numéro de référence pour trouver le prix de la pièce à partir d'une liste de prix.

Dans cet exemple, les pièces détachées d'automobile correspondent aux items de données dans la base de données. L'employé, les catalogues, les listes, les rayons du magasin, etc., correspondent au système de gestion de base de données. Pour utiliser ce « système de gestion de pièces détachées d'automobile », nous demandons à l'employé dans un langage qu'il comprend (le français) en utilisant le vocabulaire de la mécanique. « Je voudrais un carburateur pour une Renault 16 TS ». L'employé s'occupera de tout le travail d'obtention des pièces, de mise à jour des livres, et du comment utiliser ces livres, etc. Tout ce que vous avez à faire c'est d'avoir une bonne idée de ce que vous voulez. L'employé, ses livres et ses catalogues s'occupent du reste.

C'est exactement la même chose dans un système de gestion de base de données d'ordinateur (SGBD). Dès qu'un SGBD est installé sur un ordinateur, l'ordinateur devient expert dans tous les détails impliqués par le stockage, la référence, et la recherche de données. Tout ce que vous avez à faire est d'avoir une bonne idée de ce que vous voulez et de savoir un peu de vocabulaire informatique. Ce livre

vous fournira le vocabulaire informatique dont vous avez besoin. L'ordinateur et le système de gestion de base de données feront le reste.

Incidemment, ne soyez pas effrayé si vous avez à installer le système de gestion de base de données sur l'ordinateur. Ce n'est pas plus difficile que les jouets pour enfants prêts à assembler qui sont offerts en cadeaux. C'est très simple.

Un autre exemple d'un système de gestion de base de données s'apparentrait à une bibliothèque importante. Dans les grosses bibliothèques, particulièrement les bibliothèques de recherche des universités, on n'a pas accès directement aux rayons où sont rangés les livres. Pour obtenir un livre, nous devons consulter un catalogue, copier l'information de la carte d'index sur une petite feuille de papier, et porter ce papier au bibliothécaire.

Ensuite un « gnome » que vous ne voyez pas se fraye un passage dans des couloirs sombres pour retrouver le livre et le donner au bibliothécaire qui vous le remettra.



Lorsque vous rendez le livre, la plupart de ce processus est inversé. Le bibliothécaire donne le livre au « gnome » qui replace le livre à l'endroit qu'il occupait. De la même façon, le catalogue des fiches, le bibliothécaire, les « gnomes », et les possibilités de stockage correspondent à un système de gestion de base de données d'ordinateur. Les livres correspondent aux items de données. Tout ce que l'on vous demande est de savoir comment utiliser ce système. Le système fera tout le travail.

NOTRE PREMIERE BASE DE DONNEES INFORMATIQUE

Nous avons parlé de notre répertoire téléphonique personnel. C'est un bon départ pour en faire notre première base de données informatique. Nous l'utiliserons comme exemple et, lorsque nous la ferons fonctionner, vous verrez que le processus n'est pas très différent de la frappe d'informations sur une feuille de papier avec une machine à écrire ordinaire.

J'ai dit que nous tapions cette information sur une feuille de papier. D'habitude, toute personne qui sait taper à la machine doit penser à trois choses :

1) taper les titres des colonnes tels que Nom, Adresse et Numéro de téléphone ;
2) calculer le nombre d'espaces pour chaque colonne pour que tout soit net et bien en ordre ;
3) taper le titre de la page.

Dans une base de données d'ordinateur, ces opérations ne sont pas facultatives.

- Vous devez déclarer des NOMS DE RUBRIQUES pour chaque colonne (RUBRIQUE).
- Vous devez calculer la taille de chaque colonne — qui sera appelée la TAILLE DE LA RUBRIQUE.
- Et vous devez donner le titre de la base de données. Le titre de la base de données est appelé un NOM DE FICHIER.

Nous sommes pratiquement prêt à commencer la création de notre version informatique du répertoire personnel de téléphone. Mais avant d'aller plus loin, nous devons savoir que les ordinateurs ont certaines limitations.



Si vous prenez un crayon et un papier et faites la liste des noms de vos amis, leurs adresses et leurs numéros de téléphone, vous donnerez un titre à votre liste tel que « répertoire téléphonique », « liste de téléphone », etc. Avant que l'ordinateur n'accepte votre liste, vous devez lui donner un titre. Ce titre sera le nom de la base de données.

Généralement, une base de données est appelée un fichier et le titre représente le NOM DE FICHIER. Les NOMS DE FICHIERS ont certaines particularités.

- Ils n'auront pas plus de HUIT lettres et chiffres.
- Ils ne pourront contenir aucun espace vierge ou caractère réservé.
- Ils devront commencer par une lettre.

De plus, le NOM DE FICHIER est normalement précédé par un symbole supplémentaire. Ce symbole identifie le lecteur de disque qui sera utilisé pour le fichier par l'ordinateur. Un lecteur de disque est un dispositif qui est couramment utilisé pour stocker de l'information. Les dispositifs de stockage de l'ordinateur tels que les lecteurs de disque seront expliqués en détail au Chapitre 3.

Il y a souvent plus d'un lecteur de disque connecté à l'ordinateur. L'identificateur de disque est la méthode permettant d'informer l'ordinateur du lecteur qu'il pourra utiliser. Dans ce texte, les lecteurs de disque sont identifiés par une lettre suivie par deux points, tel que « A : ». C'est une convention qui est utilisée par beaucoup de systèmes micro-ordinateurs.

Quelques titres possibles (NOMS DE FICHIERS) pour notre répertoire téléphonique informatique (base de données) sont montrés à la Figure 1-3. Rappelez-vous que vous pouvez donner n'importe quel nom à la base de données avec 8 lettres ou moins.

```
                                   A:TELEPHON
(pour le lecteur de disque A)      A:LISTTEL
                                   A:ANNUAIRE

                      OU

                                   B:TELEPHON
(pour le lecteur de disque B)      B:LISTTEL
                                   B:ANNUAIRE
```

Figure 1-3. Exemple de noms de bases de données

Les noms montrés dans l'exemple ci-dessus sont appelés des mnémoniques. Un mnémonique n'est pas réellement un mot mais un groupe de lettres qui ont une prononciation phonétique qui permet de leur attribuer une signification. Une bonne habitude est de choisir des noms de fichiers de cette manière. Les cinq mots « qui suivent » sont des exemples de mnémoniques qui pourraient être utilisés comme noms de fichiers.

COMPTA PAYPERS FACTURAT PERSONNL REPORTVA

Nom de fichier

Les titres de colonnes (NOMS DE RUBRIQUES) doivent être déclarés pour chaque colonne (RUBRIQUE) dans la base de données. Les NOMS DE RUBRIQUES ont des limitations similaires aux NOMS DE FICHIERS.

- Ils n'auront pas plus de DIX lettres et chiffres.
- Ils ne peuvent pas contenir d'espaces.
- Ils doivent commencer par une lettre.

Notre simple liste de téléphone a trois titres de colonnes (NOMS DE RUBRIQUES) : NOM, ADRESSE et NUMERO TELEPHONE. Les deux premiers (NOM, ADRESSE) sont utilisables comme des NOMS DE RUBRIQUES, mais NUMERO TELEPHONE ne l'est pas. Il a plus de dix lettres et contient un espace qui n'est pas une lettre ou un chiffre. Donc, vous devez nommer cette colonne avec un mnémonique tel que NOTEL ou bien TELEPHONE ou bien NUMERO pour se conformer à la règle des 10 caractères.

Type de rubrique

Il est également nécessaire d'indiquer à l'ordinateur combien de caractères (lettres, nombres, espaces et autres symboles) seront nécessaires pour chaque RUBRIQUE. Nous taperons un nombre égal au total de positions nécessaires pour contenir les caractères, espaces et autres symboles.

Puisque l'ordinateur se comporte différemment avec les différentes sortes de rubriques, il aura besoin de connaître laquelle des trois rubriques disponibles est appropriée à ses besoins. Les rubriques peuvent être de trois sortes :

CARACTERE

NUMERIQUE, ou

LOGIQUE.

Dans notre exemple, toutes les RUBRIQUES sont des RUBRIQUES CARACTERES et peuvent contenir des lettres, des nombres, des espaces et autres symboles d'une machine à écrire standard. Les rubriques NUMERIQUES et LOGIQUES seront expliquées en détail dans les prochains exemples.

L'ordinateur fait un certain nombre de choses de sa propre initiative. Chaque fois qu'un enregistrement est saisi, l'ordinateur lui attribue automatiquement un numéro. Il appelle la première ligne, qui est le premier enregistrement, ENREGISTREMENT 1, le second ENREGISTREMENT 2 et ainsi de suite. Cela est peut-être intéressant, mais vous pourriez vous demander quelle en est la raison ? Vous pourrez également travailler avec des bases de données pendant assez longtemps et faire un certain travail avec, sans utiliser une seule fois un numéro d'enregistrement. Cependant, ils sont pratiques dans certains cas et nous discuterons de leur utilité un peu plus tard dans ce livre.

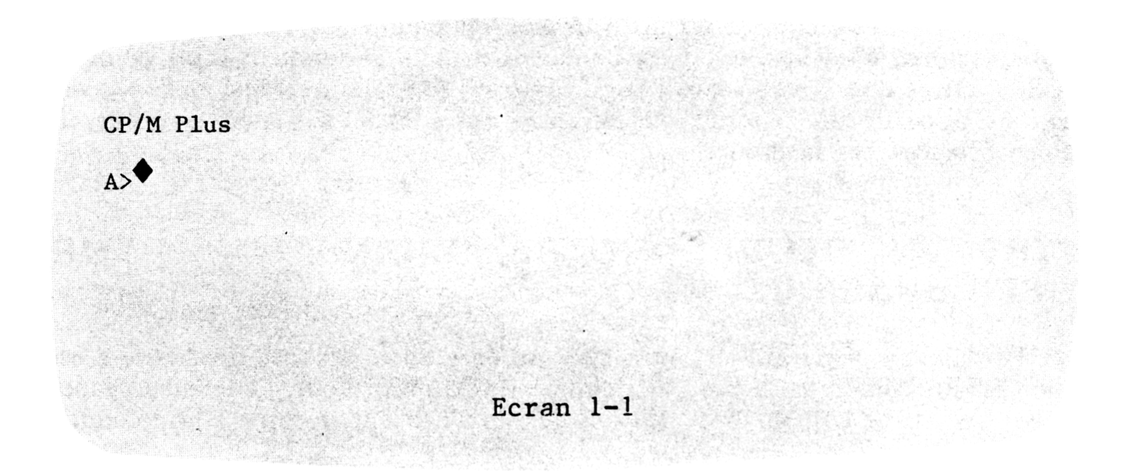
Nous venons d'étudier quelques mots du vocabulaire informatique nécessaires pour la suite. Assurez-vous d'avoir bien compris ces concepts car nous allons construire une base de données simple. Rappelez-vous :

- Les lignes et les colonnes, sont numérotées automatiquement par l'ordinateur.
- Les colonnes sont des rubriques : elles nécessitent des titres appelés des NOMS DE RUBRIQUES (10 caractères maximum, pas d'espaces, commençant par une lettre).
- Nous devons donner un titre à notre base de données, en d'autres termes un NOM DE FICHIER (8 caractères maximum, pas d'espaces, commençant par une lettre).
- Nous devons déclarer à l'ordinateur à quel endroit il doit mettre les données (quel lecteur de disque) (Exemple : A :LISTTEL les mettra sur le lecteur « A »).
- Déclarer à l'ordinateur la forme et la manière dont les informations seront saisies. Nous connaissons le NOM DE FICHIER et les NOMS DE RUBRIQUES. On termine l'opération par :

1) allouer le nombre d'espaces nécessaires pour placer l'information dans chaque rubrique, et
2) déclarer à l'ordinateur si la rubrique est du type caractère, numérique ou logique.

NOTRE PREMIER EXERCICE

Dans ce premier exercice, nous sortons réellement de l'improvisation. Lorsque l'on vient d'allumer l'ordinateur, l'écran vidéo affiche quelque chose comme la Figure 1-1.



Ecran 1-1

LE SYSTEME D'EXPLOITATION

*CP/M Plus 2.2 est un « système d'exploitation » largement utilisé sur les micro-ordinateurs. Un système d'exploitation vous aide à faire fonctionner l'ordinateur. CP/M signifie Contrôle Programme pour Micro-ordinateurs. Les systèmes de gestion de base de données commercialement disponibles tels que dBASE II utilisent le système d'exploitation pour exécuter des tâches routinières. Il n'est pas nécessaire d'apprendre tous les détails du système d'exploitation pour utiliser un système de gestion de base de données.

Le système d'exploitation impose quelquefois de faire les choses de manière particulière sur un système de base de données. Tous les exemples qui sont dans ce livre utiliseront le système des conventions d'utilisation du système d'exploitation CP/M. CP/M est un produit de Digital Research, Californie.

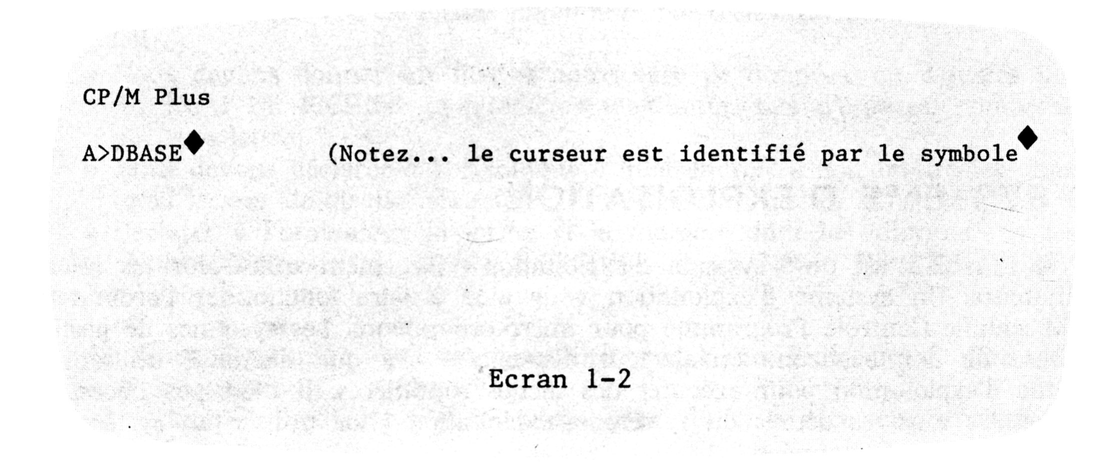
*CP/M Plus est une marque déposée de Digital Research, Inc.

Le A> est une SUGGESTION. Le système d'exploitation utilise ce moyen pour vous dire « Je suis prêt. Dites-moi ce que je dois faire. » Le symbole à droite du signe >

est appelé le curseur. Le curseur est un petit trait clignotant qui apparaît sur l'écran pour vous montrer où vous êtes. C'est l'équivalent pour l'ordinateur d'un point de crayon. Les lettres que vous « tapez » sur l'écran en pressant les touches du clavier apparaîtront à partir de l'endroit du curseur. Après chaque frappe de lettre, le curseur se déplace vers la droite.

METTRE EN OEUVRE VOTRE ORDINATEUR POUR UTILISER dBASE II

Préparer l'ordinateur pour utiliser votre système de gestion de base de données est appelé le CHARGEMENT de la base de données (LOADING). Pour commencer, tapez simplement les lettres DBASE. La vidéo affichera maintenant quelque chose comme l'Ecran 1-2.



Ecran 1-2

Appuyez sur la touche RETURN. La touche RETURN est intéressante. Le nom vient de la contraction de carriage return (Retour du chariot) d'une machine à écrire. Pour les opérations d'ordinateur, la touche peut également être gravée ENTER ou ACCEPT, ou dans des termes semblables. Pour la plupart des opérations d'ordinateur, elle lui signifie « O.K. J'y suis, je viens d'entrer cette information — maintenant fais-le ».

L'ordinateur répondra comme il est montré à la Figure 1-3 :

```
CP/M PLUS

A>DBASE

ENTREZ LA DATE DU JOUR SOUS FORME JJ/MM/AA (SINON RETOUR) : ◆
```

Ecran 1-3

Le curseur est positionné pour vous permettre d'entrer la date comme il est demandé. Entrez la date de cette manière. L'affichage vidéo apparaîtra maintenant comme illustré par l'Ecran 1-4. Notez que l'affichage commence en haut de l'écran et progresse vers le bas.

```
CP/M PLUS

A>DBASE

ENTREZ LA DATE DU JOUR SOUS FORME JJ/MM/AA (SINON RETOUR) :

***DBASE II            VERSION 2.4

 ◆
```

Ecran 1-4

Le point à l'extrémité gauche de l'écran, sous les astérisques est très important. Avec dBASE II, il est appelé POINT DE SUGGESTION. Le point de suggestion est la manière dont l'ordinateur vous indique : « Je suis prêt, dites-moi ce que je dois faire ». Les choses que vous lui direz de faire seront appelées les COMMANDES.

Pour sortir de dBASE II, vous tapez simplement le mot QUIT après le point de suggestion.

```
QUIT
```

L'ordinateur répondra par :

```
* FIN DU TRAITEMENT dBASE II version L.C.E. ***

A>
```

Vous êtes maintenant revenu au système d'exploitation de l'ordinateur.

La ligne qui suit celle où l'on demande la date vous dit exactement ce que « vous avez installé ». dBASE II est le nom du système de gestion de base de données, « Version 2.4 » vous renseigne sur la version de dBASE II que vous utilisez et la date qui suit est la date de dernière modification de la Version 2.4. Un numéro de version est comme l'identification d'un modèle. Habituellement, chaque nouvelle version contient les améliorations des versions antérieures, avec en plus des caractéristiques nouvelles et améliorées.

Vous êtes maintenant prêt à commencer le processus de CREATION de la base de données B :LISTTEL. Le dialogue entre l'utilisateur et l'ordinateur est présenté par l'Ecran 1-5. Les réponses de l'ordinateur (SUGGESTIONS) et les saisies correspondantes au clavier sont montrées comme elles apparaissent sur le terminal vidéo. Les messages de l'ordinateur sont en caractères normaux. Les entrées au clavier sont en caractères gras. Sur l'écran vidéo, l'entrée au clavier apparaît en intensité ou « brillance » normale tandis que les messages de l'ordinateur apparaissent en brillance réduite. Notez que ce que nous avons étudié tout à l'heure se retrouve maintenant sur l'écran.

```
CP/M Plus

A>DBASE

ENTREZ A DATE DU JOUR JJ/MM/AA (SINON RETOUR): 03/01/86

DBASE II            VERSION 2.4

 CREATE
ENTREZ LE NOM DE FICHIER : B:LISTTEL
ENTREZ LA STRUCTURE DE L'ENREGISTREMENT SELON LE FORMAT :
 CHAMP    NOM,TYPE,DIMENSION,DECIMALE(S)
 001      NOM,C,20
 002      ADRESSE,C,40
 003      TELEPHONE,C,8
 004

VOULEZ-VOUS COMMENCER LA SAISIE (Y/N) : Y◆
```

Ecran 1-5

Maintenant, nous entrerons les informations qui indiqueront à l'ordinateur la forme et la disposition des éléments de la base de données.

Tout de suite après le point, tapez le mot CREATE. Ensuite, appuyez sur la touche RETURN. CREATE est la COMMANDE dBASE II qui démarre le processus de construction de la base de données. RETURN indique à l'ordinateur d'accepter la COMMANDE que vous venez de taper (CREATE). Rappelez-vous que l'ordinateur ne fait rien tant que vous n'avez pas pressé la touche RETURN.

Après avoir entré le mot CREATE (et pressé la touche RETURN), l'ordinateur répondra par le message ENTREZ LE NOM DE FICHIER :. La syntaxe correcte au clavier est — l'identificateur du lecteur de disque, deux points, et les huit lettres

(et chiffres) du nom de fichier. Dans notre exemple, la syntaxe au clavier est B :LISTTEL.

Le message suivant de l'ordinateur vous demande d'entrer le nom de rubrique, le type de la rubrique et sa taille (nombre de caractères souhaités). Il affiche :

```
ENTREZ LA STRUCTURE DE L'ENREGISTREMENT SELON LE FORMAT :
CHAMP    NOM,TYPE,DIMENSION,DECIMALE(S)
001
```

Rappelez-vous que le nom de fichier peut avoir jusqu'à dix lettres et chiffres commençant par une LETTRE. Un nom de fichier ne doit pas contenir d'espaces.

La première rubrique est le champ caractère dont le NOM DE RUBRIQUE est NOM. Nous pourrions décider d'utiliser 20 caractères pour cette rubrique. Entrez au clavier NOM,C,20. Pressez alors la touche RETURN. Notez que le message de l'ordinateur demande également le nombre de positions décimales. Il n'est pas nécessaire de taper quelque chose parce que la rubrique est du type caractère. Pour les rubriques caractères, nous passons sur la demande de l'ordinateur du nombre de positions décimales. La lettre majuscule C entre les deux virgules déclare à l'ordinateur que cette rubrique est une rubrique caractère. Les virgules sont nécessaires entre les noms de rubrique, le type de rubrique, la taille et le nombre de décimales (s'il y en a).

L'ordinateur affiche également un numéro de rubrique 001 comme message. C'est exactement la même chose que lorsque l'ordinateur numérote les enregistrements. Ceci veut dire que l'information que vous venez d'entrer est la description de la première rubrique.

Lorsque la touche RETURN est pressée, après NOM, C, 20 l'ordinateur répond immédiatement par 002. Ceci veut dire qu'il est maintenant prêt à accepter le nom de rubrique, le type de la rubrique et la taille de la deuxième rubrique. Ce processus se poursuit jusqu'à l'entrée des 32 descriptions de rubriques (au maximum avec dBASE II) ou jusqu'à ce que vous ayez décidé de terminer le processus. Ne vous faites pas de souci si vous songez être obligé d'utiliser plus de 32 noms de rubriques — c'est possible mais plus complexe. Vous apprendrez un peu plus tard comment se servir de plus de 32 rubriques si c'est l'exigence de votre base de données.

Le processus peut s'interrompre à n'importe quel moment en pressant la touche RETURN au lieu d'entrer la définition de la rubrique. Dans l'exemple, RETURN est

pressé lorsque l'ordinateur répond à la définition de la rubrique 004. C'est parce que nous utilisons seulement trois rubriques dans cette base de données (NOM, ADRESSE, TELEPHONE).

LA SAISIE DES DONNEES

L'ordinateur vous demande maintenant si vous voulez entrer des informations. Vous pouvez répondre par Y pour yes ou N pour non. Il n'est pas nécessaire d'appuyer sur la touche RETURN ; l'ordinateur répond directement au Y ou N dans ce cas.

La réponse yes aura pour effet d'effacer l'écran vidéo et de vous suggérer par des messages d'entrer les noms, adresses et numéros de téléphone (informations). Entrez « Y » et l'écran vidéo vous apparaîtra tel que l'Ecran 1-6.

```
ENREGISTREMENT # 00001

NOM            :◆
ADRESSE        :
TELEPHONE      :
```

Ecran 1-6

L'affichage généré par l'ordinateur se présente en brillance normale. Les deux points indiquent le commencement et la fin de l'espace alloué à la rubrique dont le nom de rubrique apparaît à l'extrême gauche des deux points.

Cette sorte d'affichage est le mode plein écran. Il vous incite à fournir toute l'information nécessaire sur l'écran entier (contrairement au point qui vous suggère d'entrer quelque chose sur une seule ligne). L'ordinateur affiche les noms de rubriques et un espace dans lequel il attend que vous saisissiez l'information correspondante à la rubrique (colonne).

Lorsque vous entrez, par exemple, des caractères dans la rubrique nom, le curseur se déplacera à droite jusqu'à la fin de la zone. A cet instant, un bip retentit et le curseur saute à la position la plus à gauche de la rubrique suivante. Si, comme dans la plupart des cas, le nom ne remplit pas complètement l'espace qui lui est alloué, vous devez presser la touche RETURN dès que celui-ci est tapé. Suite à cette action, le curseur se déplacera à la rubrique suivante (ici, la rubrique ADRESSE).

Lorsque vous avez entré toutes les informations de l'enregistrement, comme présenté par l'Ecran 1-7, l'ordinateur effacera automatiquement l'écran et vous affichera les messages pour vous suggérer d'entrer les données de l'enregistrement suivant. Ce processus se répète jusqu'à la fin de la saisie de toutes les données.

```
ENREGISTREMENT # 00001

NOM            : BOULANGER ROBERT
ADRESSE        : 13 RUE DE LA TERASSE
TELEPHONE      : 45657800
```

Ecran 1-7

Nous avons identifié précédemment les rubriques comme des colonnes verticales. Cette fois-ci, les données sont à rentrer sur des lignes consécutives à l'écran. C'est une simple raison de commodité dans la saisie des données. L'ordinateur place verticalement chaque information de rubrique dans la base de données. Après avoir entré les données des huit enregistrements de notre exemple, l'écran ressemble à l'Ecran 1-6 mis à part le numéro d'enregistrement qui affiche 0009 ; il attend le neuvième enregistrement. Mais nous ne désirons pas en saisir un neuvième pour le moment.

Maintenant, si nous appuyons sur la touche RETURN — le curseur positionné comme le montre la figure — l'ordinateur sortira de la fonction CREATE, et répondra par un POINT, indiquant qu'il est prêt à accepter de nouvelles commandes.

L'UTILISATION DE LA BASE DE DONNEES

Nous avons maintenant une base de données. Elle a huit enregistrements et trois rubriques : NOM, ADRESSE, TELEPHONE. Son NOM DE FICHIER (titre) est LISTTEL ; il se trouve sur le lecteur B. Pour utiliser cette base de données, nous tapons au clavier USE B :LISTTEL (et RETURN) après un POINT.

```
.USE B:LISTTEL
```

L'ordinateur répond avec un autre POINT sur la ligne suivante. Vous pourriez avoir plusieurs bases de données disponibles sur votre ordinateur. Par « B :LISTTEL », vous déclarez à l'ordinateur que vous voulez travailler particulièrement avec la base de données située sur le drive B, dont le NOM DE FICHIER est LISTTEL. USE est la COMMANDE dBASE II, exactement identique lorsque vous dites à votre collaborateur, « S'il te plait Martin, passe-moi l'annuaire ». La transaction apparaît ainsi :

```
.USE B:LISTTEL
.◆
```

Lorsque l'ordinateur répond avec le POINT sur la ligne juste en-dessous USE B :LISTTEL, la base de données LISTTEL est prête à l'utilisation.

Examinons maintenant la structure que nous venons de mettre en place. La structure de la base de données est réellement déterminée par la définition des RUBRIQUES (colonnes). Pour revoir la structure, tapez DISPLAY STRUCTURE après le point. L'ordinateur répond par l'affichage présenté à l'Ecran 1-8.

```
.DISPLAY STRUCTURE                                   (Commande clavier)

STRUCTURE DU FICHIER : B:LISTTEL.DBF
NOMBRE D'ENREGISTREMENTS : 00008
DATE DE LA DERNIERE MISE A JOUR : 15/01/86
BASE DE DONNEES PRIMAIRE EN COURS D'UTILISATION
CHAMP    NOM         TYPE    TAILLE   DEC   (Réponse de l'ordinateur)
001     NOM           C         020
002     ADRESSE       C         040
003     TELEPHONE     C         008

**TOTAL**                       00069
```

Ecran 1-8

Vous pouvez remarquer que l'ordinateur a ajouté .DBF au NOM DE FICHIER de la base de données. Notez également que le « TOTAL » est un renseignement supplémentaire correspondant au nombre total de caractères utilisés dans l'enregistrement, plus 1.

.DBF est un « type de fichier ». Pour les autres fichiers que nous rencontrerons plus tard, l'ordinateur ajoute également le type de fichier. Ce dernier renseigne l'ordinateur sur la manière de traiter le fichier. La date de la dernière mise à jour est celle qui a été tapée lorsque nous avons mis en route dBASE II. A chaque mise en route, une date est tapée au clavier. Si une modification intervient dans la base de données, la date de la dernière mise à jour sera modifiée. Si vous répondez par RETURN — une autre façon d'entrer une date — la date de la dernière mise à

jour se présentera comme 00/00/00. L'élément « total » qui apparaît en base indique le nombre de caractères (plus 1) de l'enregistrement. C'est l'indicateur de taille d'enregistrement et sa signification sera expliquée un peu plus tard.

Maintenant que nous avons utilisé (USE) B:LISTTEL pour voir la structure que nous venons de créer, passons à une utilisation plus sérieuse des bases de données : le stockage et la recherche d'information. Nous voulons retrouver quelques informations que nous avons stockées de façon à les examiner. La manière de voir l'information est de taper DISPLAY ALL après le POINT. DISPLAY ALL est la commande dBASE II qui affiche le contenu de la base de données. Le résultat de cette commande est montré par l'Ecran 1-9.

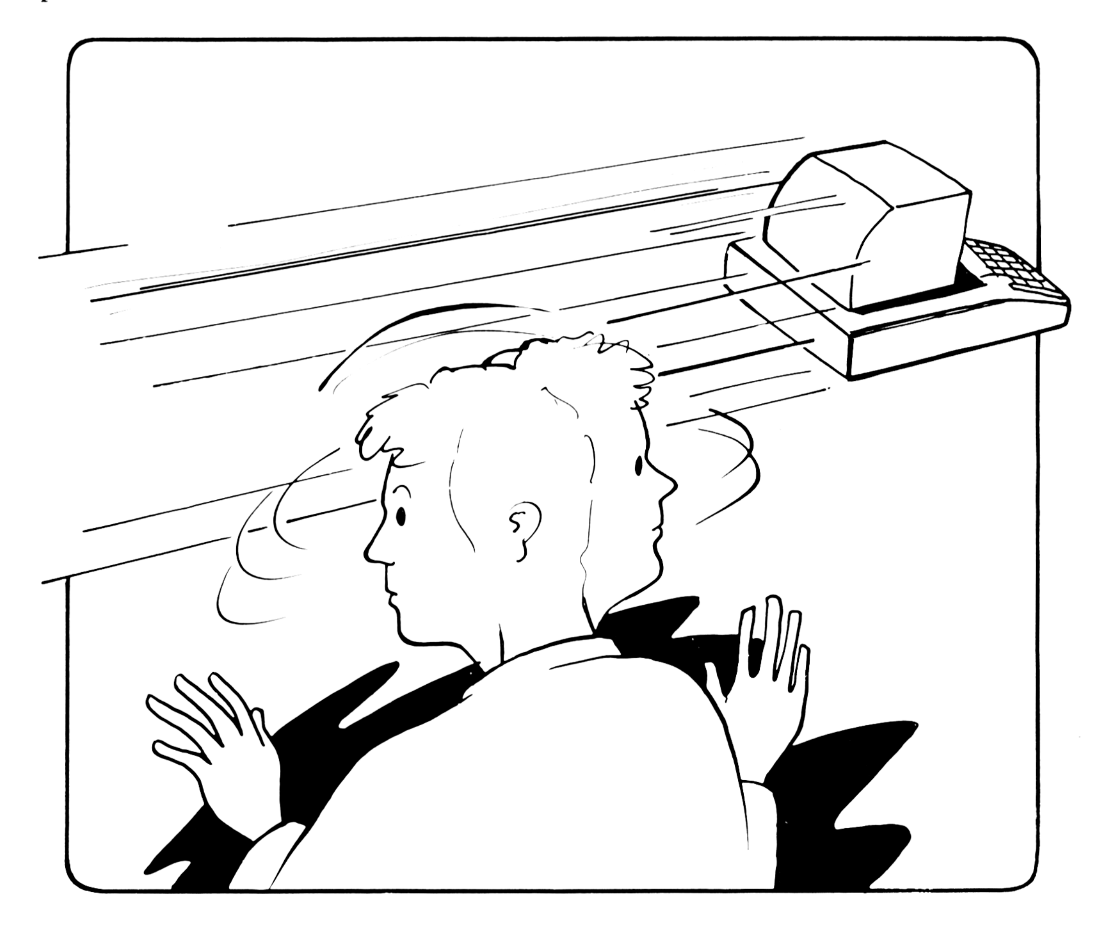
```
.USE B:LISTTEL
.DISPLAY ALL

00001    Boulanger Robert        13 Rue de la Terasse       45657800
00002    Fouquet Sébastien       69 Rue de la Trésorerie    41122830
00003    Lamotte Robert          4 Rue de la Laiterie       45583244
00004    Legendre Jacques        125 Rue des Ormes          42283042
00005    Lenoir Christian        11 Rue du Cimetière        47712139
00006    Robertson Stanislas     506 Avenue Foch            43334312
00007    Poupon Pierre           4 Boulevard du Temple      47011359
00008    Schmitt Wilfrid         41 Rue de la Canebière     48083216
```

Ecran 1-9

La colonne la plus à gauche affiche le numéro d'enregistrement affecté automatiquement comme nous l'avons vu. Si vous ne voulez pas voir affiché le numéro d'enregistrement, tapez DISPLAY ALL OFF à la place de DISPLAY ALL. L'ordinateur présentera le même affichage sans les numéros d'enregistrements.

Nous avons donné, dans ce chapitre, quelques exemples de ce que peut facilement faire l'ordinateur. L'un d'eux utilisait un numéro de téléphone sur un morceau de papier. La commande DISPLAY de dBASE II vous affichera le nom du propriétaire de ce numéro (s'il se trouve dans la base de données). Pour arriver à ce résultat, nous devons dire à l'ordinateur ce que nous voulons faire d'une manière compréhensible. L'ordinateur peut être rapide, mais il n'est pas particulièrement perspicace.



Donc, nous utiliserons la commande DISPLAY de dBASE II pour récupérer le « propriétaire » du numéro de téléphone sur le bout de papier que nous avions ramassé. La transaction apparaît comme suit :

```
.DISPLAY FOR TELEPHONE = "42283042"

00004   Legendre Jacques    125 Rue des Ormes      42283042
```

N'est-ce pas surprenant ? Le mot TELEPHONE est le NOM DE RUBRIQUE de la RUBRIQUE (colonne) contenant les numéros de téléphone. Vous êtes réellement en train de demander à l'ordinateur avec la courte instruction suivante : « Examinez toute la base de données et affichez chaque enregistrement contenant les caractères '42283032' en colonne téléphone.

Les apostrophes à chaque extrémité du numéro de téléphone sont importantes. Si elles ne sont pas présentes, l'ordinateur ne sait que faire sauf vous indiquer que vous avez commis une erreur.

Ces apostrophes sont appelées des DELIMITEURS. L'ordinateur s'en sert pour identifier le début et la fin de ce qui est appelé une « CHAINE DE CARACTERES ». Ce qui est contenu entre apostrophes est spécifiquement ce que vous attendez (lettre et espace).

Les numéros de téléphone sont des chaînes de caractères parce que nous identifions la rubrique numéro de téléphone (TELEPHONE) comme une rubrique caractère, lorsque nous avons créé la base de données. Les nombres peuvent être soit des valeurs numériques soit des caractères. Nous devons donc indiquer à l'ordinateur quel type nous avons choisi en utilisant les délimiteurs. Comme exemple, 55 est un nombre pour l'ordinateur, tandis que « 55 » est une chaîne de caractères. Les nombres doivent être entrés habituellement comme des caractères sauf s'ils sont susceptibles d'être utilisés pour des calculs arithmétiques.

Comme autre exemple, cherchons une personne qui habite Rouen.

```
.DISPLAY FOR 'ROUEN'$ADRESSE

00006  Robertson, Stanislas   506 Avenue Foch, ROUEN          43334312
00002  Fouquet, Sébastien     69 Rue de la Trésorerie, ROUEN 41122830
```

De la même façon que tout à l'heure, la chaîne de caractères est mise entre apostrophes. Cet exemple, cependant, apparaît un peu spécial. Le signe $ est un symbole qui veut dire « contenu dans ». Ce que vous dites à la machine peut s'exprimer ainsi : Examinez toute la base de données et affichez chaque enregistrement qui contient les caractères 'Rouen' dans la colonne Adresse. S'il existe une Avenue de Rouen, quelque part dans cette rubrique de la base de données, cet enregistrement apparaîtra avec les deux autres. Finalement, ce que l'ordinateur examine c'est une séquence de caractères. Pour illustrer ce point, indiquons à l'ordinateur de :

```
DISPLAY FOR 'ROUEN'$ADRESSE
•
```

L'ordinateur répond par un POINT. Ceci veut dire qu'il n'y a pas d'enregistrement avec la chaîne de caractères ROUEN dans la rubrique adresse. Les deux chaînes de caractères Rouen et ROUEN sont différentes. La première utilise les minuscules tandis que l'autre utilise les majuscules. Dans cet exemple, nous savons vous et moi qu'il s'agit de la même chose, mais pas pour l'ordinateur. L'interprétation .de l'ordinateur est littérale.

Un autre exemple d'interprétation littérale de l'ordinateur : demandons à l'ordinateur d'afficher tous les enregistrements contenant la CHAINE DE CARACTERES « Robert ». La réponse de l'ordinateur est représentée par l'Ecran 1-10.

```
.DISPLAY FOR 'Robert'$NOM

00001    Boulanger Robert       13 Rue de la Terasse      45657800
00003    Lamotte Robert         4 Rue de la Laiterie      45583244
00006    Robertson Stanislas    506 Avenue Foch           43334312
```

Ecran 1-10

Ceci est un exemple graphique de la recherche d'une chaîne de caractères où l'ordinateur donne plus d'informations que nous lui en avions demandé. En fait ici : il a fait exactement ce qu'on lui a dit. Dans ce cas, nous voulions un affichage de toutes les personnes qui se nommaient Robert. Ceci aurait pu être obtenu en utilisant ',Robert' à la place de 'Robert' dans l'instruction. 'Robert' comme prénom est toujours précédé d'une virgule.

Pour démythifier un peu (s'il y a un mythe) les opérations sur les bases de données d'ordinateur, examinons ce qui se passe lorsque nous disons DISPLAY FOR 'ROUEN'$ADRESSE.

- L'ordinateur « se place » au début de la base de données (Enregistrement 1) et examine la rubrique ADRESSE de l'Enregistrement 1 pour voir si la CHAINE DE CARACTERES 'ROUEN' s'y trouve.

- Si elle s'y trouve, l'ordinateur affiche tout l'Enregistrement 1 sur le terminal vidéo.

- Si elle ne s'y trouve pas, l'ordinateur n'affiche rien de l'Enregistrement 1.

- Lorsque l'Enregistrement 1 a été examiné et affiché ou non sur le terminal, l'ordinateur se met à examiner l'Enregistrement 2 exactement de la même façon.

- Cette inspection enregistrement par enregistrement de la RUBRIQUE ADRESSE continue jusqu'à ce que chaque enregistrement ait été examiné.

Récapitulons ce que nous venons de faire. Nous venons d'apprendre un peu de vocabulaire, que les bases de données peuvent être rapprochées de choses que nous utilisons tous les jours, et que les bases de données et les ordinateurs peuvent faire des choses que nous ne pouvons pas faire sans eux — sauf si nous avions énormément de temps.

Jusqu'à maintenant, un papier et un crayon auraient été beaucoup plus rapides, moins chers et vous n'auriez pas eu besoin d'apprendre un vocabulaire informatique. Ayez quand même à l'esprit que cette liste pourrait être facilement longue de 80, 800 ou 8.000 noms. Les exemples dans ce livre n'ont jamais plus de 15 enregistrements (dans l'intérêt de conservation du papier et votre intérêt). De façon à apprécier ce que la technologie des ordinateurs peut faire pour vous, essayez d'imaginer ces exemples comme étant des parties de bases de données contenant des milliers d'enregistrements.

L'ensemble du processus ressemble à la fabrication d'un tableau avec du papier et un crayon.

- Vous devez préparer vos idées, décider quelle information va dans quelle colonne, et déterminer la taille physique de chaque colonne.

- Vous entrez alors toutes les informations.

- C'est seulement après lorsque tout est terminé que vous pouvez réellement utiliser le tableau de la manière prévue. Dans ce chapitre, nous utilisons seulement la base de données pour l'examiner et l'afficher (DISPLAY).

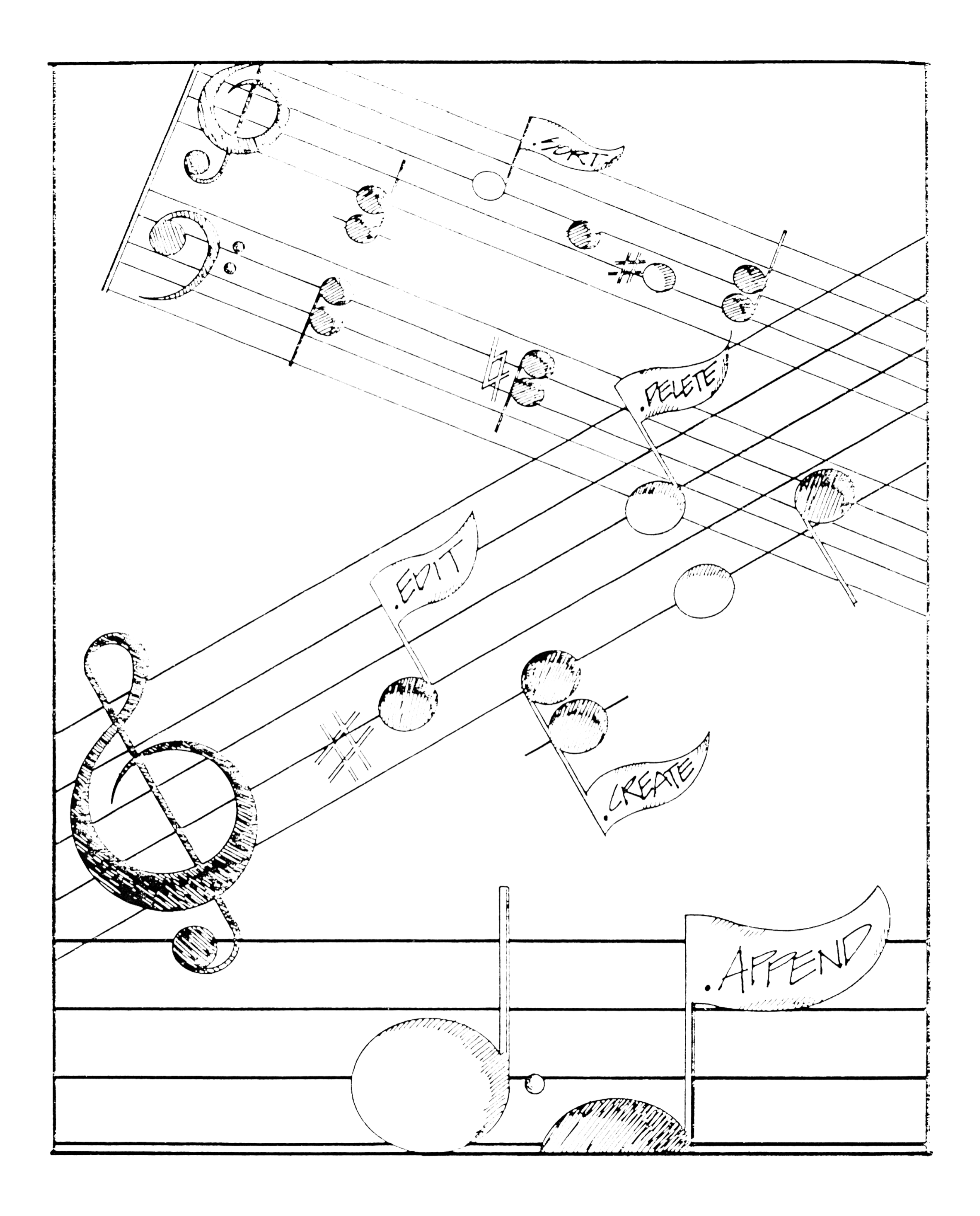
Le processus de fabrication d'un tableau sur papier ressemble exactement au processus d'une base de données d'ordinateur. La base de données doit être préparée. L'information doit être saisie. Ensuite, et seulement ensuite, vous pouvez l'utiliser pour faire quelque chose.

L'amusant dans tout cela c'est que vous pouvez accomplir toutes ces choses sans vraiment connaître le fonctionnement interne de l'ordinateur. Tout ce que vous avez à faire, c'est de suivre les règles. C'est un petit peu comme conduire une automobile avec une boîte automatique à la place d'une voiture avec une boîte de vitesse manuelle.

Le langage que vous utilisez pour parler à l'ordinateur est un anglais très limité. Presque toutes les bases de données disponibles commercialement sur micro-ordinateur ont des vocabulaires « anglais » de moins de cent mots. Chaque mot d'anglais signifie ce qu'il signifiait la dernière fois qu'il a été utilisé. Chaque mot d'anglais dans le vocabulaire de l'ordinateur a un sens très étroit et ne laisse la place à aucune interprétation. La signification de ce mot sera UNIQUE parmi toutes les significations anglaises courantes de ce mot.

En fin de compte, vous n'avez pas à vous inquiéter. Lorsque vous apprenez, vous ne risquez pas de détruire l'ordinateur et si vous n'êtes pas sûr à propos de quelque chose — essayez autre chose et regardez si cela fonctionne. Peut-être la chose la plus difficile à apprendre c'est qu'il est simple de travailler avec l'ordinateur. C'est un outil facile à maîtriser. Nous utiliserons un vocabulaire volontairement limité et ce travail ne requiert pas une coordination physique particulière.


L'ORDI
NA
-TEUR
BRAVO!


.SORT
.DELETE
.EDIT
.CREATE
.APPEND

CHAPITRE II

L'UTILISATION D'UNE BASE DE DONNEES SIMPLE : SON FONCTIONNEMENT

Maintenant que vous êtes familiarisé avec une base de données, nous sommes prêt à étudier les différentes utilisations et le fonctionnement d'une base de données simple. Nous avons déjà vu un type d'utilisation avec DISPLAY. A cette occasion nous avons examiné la structure et le contenu d'une base de données très simple.

Dans ce chapitre, nous apprendrons à modifier la base de données — en ajoutant et supprimant de l'information, la manière de produire des documents standard, indexer et trier le contenu d'une base de données. Nous créerons également une base de données sensiblement plus complexe que B :LISTTEL du Chapitre I.

Nous avons donc étudié deux façons simples d'utiliser les bases de données :

- la manière d'examiner sa structure, et
- les manières de voir tout ou certaines parties du contenu.

La modification sera le processus suivant d'utilisation de notre base de données. Il est très important de savoir que nous pouvons facilement faire les modifications nécessaires.

Si nous reprenons l'exemple de liste de téléphones personnels, nous pouvons espérer pouvoir facilement « effacer » les noms, adresses et numéros de téléphone et en entrer de nouveaux à leurs places.

C'est effectivement le cas. Cette possibilité est nécessaire pour différentes raisons. Il arrive de faire des erreurs lorsque l'on saisit l'information — une adresse et un numéro de téléphone sont saisis avec un nom incorect, les mots sont mal orthographiés, les numéros sont oubliés, etc. Les gens se déplacent et les numéros de téléphone ou les adresses changent. D'autres se marient et leurs noms changent.

Certaines personnes disparaissent de la vie — d'autres y entrent. Si le monde était vraiment statique sans aucun changement, l'importance des bases de données serait réduite. Le monde, cependant, change à une vitesse considérable ce qui donne une certaine valeur aux bases de données.

MODIFICATIONS

Il y a généralement trois sortes de modifications opérées par l'ordinateur :

- La première est d'ajouter ou supprimer des enregistrements entiers
- La deuxième est de modifier le contenu d'un enregistrement
- La dernière est de modifier la structure d'une base de données en ajoutant ou en supprimant des rubriques entières (colonnes).

SUPPRESSION D'ENREGISTREMENTS

La suppression d'un enregistrement est un processus en deux temps. Ce qui permet une sauvegarde contre des effacements accidentels. Avec dBASE II, on a deux commandes séparées : DELETE et PACK. Pour illustrer la procédure, nous effacerons l'enregistrement de Jacques Legendre (ENREGISTREMENT 4). La transaction avec l'ordinateur est illustrée par l'Ecran 2-1.

```
.DELETE FOR NOM = "Legendre, Jacques"
00001 EFFACEMENT(S)
.DISPLAY FOR*
00004 .    *Legendre, Jacques       125 Rue des Ormes       42283042
PACK
COMPACTAGE TERMINE 00007 ENREGISTREMENTS TRANSFERES
```

Ecran 2-1

Nous demandons à l'ordinateur de supprimer l'enregistrement qui contient Jacques Legendre. La commande DELETE place une marque (*) en face de l'enregistrement que nous voulons supprimer. L'ordinateur nous indique (par un astérisque) qu'il a marqué un enregistrement pour effacement. A partir de ce moment, l'enregistrement est encore dans la base de données. S'il avait trouvé un autre Jacques Legendre, la réponse serait « 00002 EFFACEMENTS », nous sommes alors prévenu que quelque chose d'inattendu s'est produit.

La commande DISPLAY FOR indique à l'ordinateur d'afficher les enregistrements marqués pour une suppression. L'ordinateur affiche un simple enregistrement, celui marqué pour effacement. Notez que le symbole * apparaît accolé au numéro d'enregistrement.

C'est la commande PACK qui supprime l'enregistrement. Elle élimine tous les enregistrements marqués par la commande DELETE et renumérote tous les numéros d'enregistrements restants.

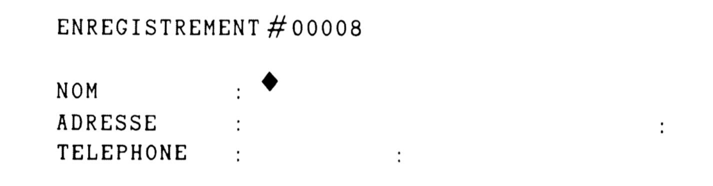
Si nous affichons maintenant toute la base de données, nous voyons que l'enregistrement de Jacques Legendre a disparu. Il y a maintenant 7 enregistrements seulement, et l'enregistrement qui était au départ numéro 5 se trouve maintenant numéro 4 et ainsi de suite. La base de données modifiée est montrée à l'Ecran 2-2.

```
00001   Boulanger Robert       13 Rue de la Terasse        45657800
00002   Fouquet Sébastien      69 Rue de la Trésorerie     41122830
00003   Lamotte Robert         4 Rue de la Laiterie        45583244
00004   Lenoir Christian       11 Rue du Cimetière         47712139
00005   Robertson Stanislas    506 Avenue Foch             43334312
00006   Poupon Pierre          4 Boulevard du Temple       47011359
00007   Schmitt Wilfrid        41 Rue de la Canebière      48083216
```

Ecran 2-2

AJOUT D'ENREGISTREMENTS (APPEND)

Ajouter un enregistrement à la base de données est un processus en une seule étape. La commande d'ajout d'enregistrement avec dBASE II s'appelle APPEND. Saisir une information avec APPEND équivaut à saisir de l'information avec CREATE. Lorsque l'on indique à l'ordinateur APPEND, celui-ci efface l'écran et génère l'affichage montré sur l'Ecran 2-3.



Ecran 2-3

Pour ajouter l'enregistrement, remplissez les blancs comme dans le processus CREATE. Cela équivaut à remplir un formulaire avec une machine à écrire. Vous pouvez ajouter autant d'enregistrements que vous désirez. Pour sortir de APPEND, appuyez sur la touche RETURN lorsque le curseur est à la première position de la première rubrique du nouvel enregistrement — exactement comme dans CREATE.

Les nouvelles données sont montrées par l'Ecran 2-4. Dans cet exemple, nous ne connaissons pas l'adresse de T.E. DEUM aussi un RETURN a été provoqué pour avancer le curseur à la rubrique TELEPHONE.

```
ENREGISTREMENT # 00008

NOM          : Deum, T.E.
ADRESSE      :
TELEPHONE    : 43339194 :
```

Ecran 2-4

Comme dans l'exemple CREATE, l'ordinateur avance automatiquement vers l'enregistrement suivant (Enregistrement 9) et vous suggère d'entrer les données. Le fait d'appuyer sur la touche RETURN arrêtera le processus APPEND et vous obtenez alors le point de suggestion.

EDIT & REPLACE

Maintenant, supposons que nous ayons une adresse pour T.E. DEUM et une modification de son numéro de téléphone. On utilisera deux commandes principales pour modifier les rubriques dans un enregistrement. Ce sont EDIT et REPLACE.

La commande EDIT est similaire à la commande APPEND excepté qu'elle n'ajoute pas d'enregistrement. Contrairement à la commande APPEND, elle nécessite la connaissance du numéro d'enregistrement. Elle fait partie des quelques commandes dBASE II qui requièrent la connaissance des numéros d'enregistrements. Un enregistrement peut être modifié (EDIT) sans connaître son numéro mais le processus s'effectue alors en deux étapes. L'utilisation de la commande EDIT est décrite ci-dessous. Pour appeler la fonction EDIT, tapez EDIT suivi du numéro d'enregistrement désiré, après le point.

```
.EDIT 8
```

EDIT est similaire à APPEND, excepté qu'à la place des rubriques vides, comme dans APPEND, est affiché le contenu des rubriques de l'enregistrement désiré. Le

curseur apparaît, comme dans APPEND, à la première position de la première rubrique. Dans cet exemple, nous ajouterons l'adresse et modifierons le numéro de téléphone.

Pour modifier l'adresse, nous devons déplacer le curseur à la rubrique ADRESSE. Il faut nous rappeler ce que vous avons vu précédemment. Le mouvement de curseur nécessite l'utilisation de la touche CONTROLE ou CTRL (CPC 6128) ou encore la touche ALT (PCW 8256).

LA TOUCHE CONTROLE

La touche CONTROLE ne se trouve pas sur une machine à écrire ordinaire. On s'en sert de la même manière que la touche SHIFT ou montée de chariot sur une machine à écrire. La touche SHIFT permet une « deuxième signification » à chaque touche de la machine. Le fait de presser la touche SHIFT pendant que l'on presse la touche 'h' par exemple fournira la lettre majuscule 'H' sur la machine. La touche CONTROLE (ou CTRL) fournit une « troisième signification » à la plupart des touches. Par exemple, lorsque nous pressons la touche CONTROLE, la touche 'X' avancera le curseur d'une rubrique chaque fois que l'on pressera celle-ci. On abrégera cette action par CONTROLE X. On écrit souvent ^X (ou CTRL X ou CODE X). Le résultat de la plupart des touches de CONTROLE est résumé par le Tableau 2-1

TOUCHE DE CONTROLE	EFFET
Ctrl - X ou ↓	Déplace le curseur d'une rubrique en avant
Ctrl - E ou ↑	Déplace le curseur d'une rubrique en arrière
Ctrl - D ou →	Déplace le curseur d'un espace en avant
Ctrl - S ou ←	Déplace le curseur d'un espace en arrière
Ctrl - Y	Efface le contenu de la rubrique
Ctrl - V	Insère un caractère à la position du curseur
Ctrl - G CLR ou DEL →	Efface un caractère à la position du curseur
Ctrl - W	Sort de la fonction EDIT

Tableau 2-1. Fonction de quelques touches de contrôle

Maintenant que nous comprenons les fonctions de quelques touches de contrôle, nous pouvons faire la modification du RECORD 8 (Enregistrement 8). Juste avant de parler des touches de contrôle, nous avions tapé EDIT 8 en réponse au POINT, le Record 8 est maintenant affiché à l'écran, avec le curseur positionné sur la rubrique NOM.

```
ENREGISTREMENT  # 00008


NOM            : Deum, T.E.
ADRESSE        : 111 Grande Rue, Paris
TELEPHONE      : 49991345 :
```

Ecran 2-5

Pour déplacer le curseur à la rubrique ADRESSE, pressez CTRL C (ˆ). Ceci aura pour effet de déplacer le curseur à la première position caractère de la rubrique ADRESSE. Entrez l'adresse comme montré par l'Ecran 2-5. Le fait de presser la touche RETURN positionnera le curseur au début de la rubrique TELEPHONE. Pour modifier le numéro de téléphone, tapez simplement le nouveau numéro directement sur l'ancien.

Lorsque le chiffre « 4 » sera tapé dans la zone numéro de téléphone, l'ordinateur sortira automatiquement de la commande EDIT. Il sort de cette commande car nous avons tapé le dernier caractère dans la dernière rubrique du dernier enregistrement de la base de données. Si cela n'avait pas été la dernière, il n'aurait pas été judicieux de permettre la sortie de cette commande. Pour en sortir, il faut alors taper CONTROLE « W » dans ce cas. ˆW indique à l'ordinateur d'« écrire » la nouvelle information ou la structure sur le disque. Le fait de « sortir » veut dire que vous avez terminé avec les modifications et que celles-ci doivent être « écrites » sur le disque pour un stockage permanent.

L'autre commande de modification, REPLACE, permet une modification sélective d'un ou plusieurs enregistrements sur des critères particuliers. Avec REPLACE, vous modifiez le contenu d'une ou plusieurs rubriques en indiquant à l'ordinateur les noms de rubriques, le nouveau contenu, et le critère à partir duquel vous voulez effectuer la modification.

Comme exemple, supposons que T.E. DEUM ait changé de numéro de téléphone et d'adresse. Nous pouvons modifier ces deux rubriques avec la commande REPLACE :

```
.REPLACE TELEPHONE WITH '42226661',ADRESSE WITH '200 Rue Lejeune,ANTONY'
FOR NOM='Deum, T.E.'
0001 REMPLACEMENT (S)
```

L'ordinateur nous répond 0001 REMPLACEMENT(S) bien que la commande ait modifié à la fois le numéro de téléphone et l'adresse par une nouvelle information.

EDIT et REPLACE ont tous les deux des utilisations extensives pour modifier le contenu d'une base de données. EDIT est une opération que l'on appelle en mode plein écran. L'enregistrement EST SUR L'ECRAN et peut être modifié en tapant la nouvelle information par dessus l'ancienne. REPLACE sert à remplacer automatiquement certaines informations par de nouvelles à partir de critères choisis par l'utilisateur. La modification est faite dans l'ordinateur mais ne se verra pas sur l'écran. Si une confirmation visuelle est souhaitée, il faut utiliser une commande d'affichage séparée.

L'exercice sur le répertoire téléphonique contient des informations personnelles qu'il est facile de retrouver car nous connaissons bien toutes ces personnes. Cependant, il est très limité. Il ne nous fournira pas suffisamment de matière pour faire une démonstration de l'étendue des possibilités d'une base de données d'ordinateur.

Donc, il est temps de préparer notre second projet de base de données. Notre nouvelle base de données sera capable d'illustrer un maximum d'aspects des possibilités d'une base de données informatique. Cette base de données nous est familière. Elle résoud les problèmes de stock d'une boutique de détail. Nous ferons un inventaire annuel similaire à celui qui pourrait être effectué par le propriétaire d'une petite boutique de vente d'alcools.

Quels seront les items qui peuvent être intéressants pour le propriétaire d'une boutique d'alcools ? Ils incluront vraisemblablement :

Le type d'alcool
Le nom de la marque
La contenance
La quantité en stock
Le prix de revient
Le prix de vente

Ces items deviennent alors les colonnes (rubriques) nécessaires dans notre nouvelle base de données. Nos trois étapes suivantes font toutes partie d'un seul processus qui définit la forme de la base et son aménagement.

1. Nous déclarons les NOMS DE RUBRIQUES
2. Nous affectons des catégories ou types à chaque rubrique (caractère, numérique, logique)
3. Nous décidons la taille (whidth) de chaque rubrique

- Dans cet exemple, le nom de rubrique de la colonne contenant le type d'alcool est ALCOOL. C'est une rubrique caractère qui fait 10 espaces de large.
- La colonne des noms de marques a le nom de rubrique MARQUE, c'est une rubrique caractère, elle fait 20 espaces de large.
- Celle des contenances est CONT, c'est une rubrique caractère et elle fait 7 espaces de large.
- Le montant du stock est QUANTITE, une rubrique numérique, de 3 positions de large. On peut placer jusqu'à 999 bouteilles de chaque contenance de chaque marque dans le stock de ce petit magasin.
- Les deux dernières rubriques, appelées PXRV et PXVT, sont seulement des rubriques numériques. Ces deux rubriques contiendront des valeurs décimales, on les saisira cependant d'une manière différente. Pour les deux, nous choisirons 3 positions à gauche du point décimal, et 2 positions à droite. Ce qui nous permet un maximum de 999.99. La taille de la rubrique de chacune de ces deux rubriques est de SIX (5 positions plus le point décimal).

Nous avons exactement le même problème pour choisir un NOM DE FICHIER (titre) à cette base de données comme nous l'avions dans l'exemple de la liste de téléphones personnels. Le titre, Inventaire d'un magasin d'alcools, est beaucoup trop large et contient des espaces blancs. Nous choisirons finalement STOCK comme nom de fichier. Il est significatif et n'a que cinq lettres.

CREATION D'UNE BASE DE DONNEES

Le processus que nous venons de voir est présenté par l'Ecran 2-6. La base de données (fichier) STOCK sera placée sur le lecteur de disque B.

```
.CREATE
ENTREZ LE NOM DE FICHIER : B:STOCK
ENTREZ LA STRUCTURE DE L'ENREGISTREMENT SELON LE FORMAT :
 CHAMP    NOM,TYPE,DIMENSION,DECIMALE(S)
 001      ALCOOL,C,10
 002      MARQUE,C,20
 003      CONT,C,7
 004      QUANTITE,N,3
 005      PXRV,N,6,2
 006      PXVT,N,6,2
 007

VOULEZ-VOUS COMMENCER LA SAISIE ? Y
```

Ecran 2-6

Avec Y (yes) comme réponse, l'ordinateur vous suggère de saisir les données du premier enregistrement. L'affichage vidéo est montré par l'Ecran 2-7.

```
ENREGISTREMENT#00001

LIQUEUR      :            :
MARQUE       :                        :
CONT         :           :
QUANTITE     :      :
PXRV         :        :
PXVT         :        :
```

Ecran 2-7

Nous entrerons maintenant les données dans la base STOCK. Un enregistrement significatif est présenté à l'Ecran 2-8.

```
ENREGISTREMENT#00001

LIQUEUR      : SCOTCH     :
MARQUE       : LONG JACK              :
CONT         : 90 CL     :
QUANTITE     :    23:
PXRV         : 39.00 :
PXVT         : 45.70 :
```

Ecran 2-8

Ce processus se continue, enregistrement par enregistrement, jusqu'à ce que toutes les données des quinze enregistrements soient saisies. Un RETURN au début de l'enregistrement 16 permet de sortir du mode saisie de données.

REMARQUES SUR LA SAISIE DE DONNEES

La saisie de données commence à gauche à partir des deux points pour chaque rubrique. Si la rubrique est du type caractère, les données resteront à l'extrémité gauche des deux points, exactement de la même façon que lorsque vous remplissez un imprimé. C'est ce que l'on appelle la *justification à gauche.*

Pour les rubriques du type numérique, les données sont saisies en commençant à gauche des deux points, exactement comme pour les rubriques du type caractère. Cependant, lorsque vous appuyez sur la touche RETURN (vous indiquez à l'ordinateur que vous venez de terminer la saisie de la rubrique), l'ordinateur déplacera les nombres à l'extrémité droite des deux points de la colonne, comme si vous écriviez des nombres dans une colonne. C'est ce que l'on appelle la *justification à droite.* L'ordinateur justifie à droite les rubriques numériques.

Si vous tentez de saisir une lettre dans une rubrique numérique, dBASE II vous l'interdira. Le caractère ne sera pas accepté et le terminal émettra un bip sonore pour vous avertir.

Le processus de saisie de données continue jusqu'à ce que le stock soit complet (ou lorsqu'il y a assez de données pour notre exemple). Votre exemple de base de données B :STOCK a 6 rubriques et 15 enregistrements.

De manière à se servir de la base de données, nous devons d'abord indiquer à l'ordinateur que nous voulons utiliser cette base. Pour ce faire, nous tapons la commande LIST B :STOCK. Le contenu complet de l'exemple B :STOCK est montré par l'Ecran 2-9.

```
.USE B:STOCK
.LIST

00001  SCOTCH    LONG JACK     90 cl      32  39.00   45.70
00002  SCOTCH    LONG JACK     1 l         7  43.00   48.30
00003  SCOTCH    LONG JACK     70 cl      88  29.80   50.10
00004  VODKA     PETROVNA      2 l        35  23.78   33.50
00005  VODKA     PETROVNA      1 l         9  27.95   37.90
00006  VODKA     PETROVNA      90 cl      75  21.49   31.30
00007  WHISKEY   4 ROSES       1 l 1/2    32  25.11   40.10
00008  WHISKEY   4 ROSES       2 l        44  31.98   41.00
00009  WHISKEY   4 ROSES       90 cl      19  18.50   28.50
00010  WHISKEY   NEW SOUTH     1/2 l       4  12.80   22.20
00011  BOURBON   SPECIAL       70 cl       5  41.78   52.25
00012  BOURBON   SPECIAL       90 cl      22  43.50   51.00
00013  BOURBON   SPECIAL       1 l        21  58.10   67.15
00014  BOURBON   SPECIAL       1 l 1/2     3  67.30   79.12
00015  BOURBON   SPECIAL       3 l         5  67.30   79.12
```

Ecran 2-9

Un des objectifs d'un inventaire est de déterminer le montant de capital immobilisé par le stock. Lorsque nous avons terminé un inventaire de façon conventionnelle, avec du papier et un crayon, nous pouvons calculer la valeur du stock en multipliant chaque quantité par le prix correspondant et en ajoutant les résultats. Si l'on gère le stock avec un micro-ordinateur et un système de gestion de base de données, tel que dBASE II, nous pouvons obtenir le même résultat presque immédiatement. Le système peut faire cela pour nous. A partir de cet exemple, nous commençons à entrevoir la puissance d'une base de données.

La commande dBASE II qui permet d'obtenir la valeur du stock se présente comme suit :

```
.SUM PXRV*QUANTITE
12395.14
```

Nous demandions la somme (total) des coûts multipliée par les quantités. Le symbole astérisque (*) représente la multiplication pour l'ordinateur. Le résultat de cette opération (12395.14) est présent juste en dessous de la commande. L'ordinateur applique la commande ci-dessus à toute la base de données.

Supposons que nous voulions savoir quel est le capital qui est investi dans le scotch. L'opération se présente comme suit :

```
.SUM PXRV*QUANTITE FOR ALCOOL='SCOTCH'
4171.40
```

Maintenant, supposons que nous voulions savoir quel est l'investissement en scotch dans la contenance 1/4 de litre. La commande et le résultat sont montrés ci-dessous. Dans cet exemple, nous voyons une autre particularité de l'ordinateur. AND doit être précédé et suivi par un point de façon qu'il se présente comme suit :

```
.SUM PXRV*QUANTITE FOR ALCOOL='SCOTCH'.AND.CONT='70cl'
2622.40
```

Des simples calculs comme ceux-là ne représentent que le début de l'information disponible à partir d'opérations sur une base de données. Nous pouvons actuellement obtenir des documents de résultats complets, organisés pour utiliser et afficher notre information à partir de nos besoins, avec uniquement deux commandes.

DOCUMENTS STANDARD

dBASE II contient un « générateur d'état » que l'on appelle REPORT. Beaucoup de documents peuvent être préparés à partir de cette facilité. Les documents peuvent être aménagés directement à partir du clavier de l'ordinateur et le format du document peut être sauvegardé pour un usage ultérieur.

Après le POINT, tapez REPORT. L'ordinateur vous posera une série de questions appelées SUGGESTIONS. C'est exactement comme si l'on remplissait un imprimé. Les réponses sont exploitées par l'ordinateur pour préparer le document.

- Le premier message est la demande du NOM DU FORMAT D'IMPRESSION. C'est une autre sorte de NOM DE FICHIER, similaire au NOM DE FICHIER utilisé pour la base de données. Les noms de fichiers des formats de documents suivent les mêmes règles que les noms de fichiers pour les bases de données. Ils doivent commencer par la lettre identifiant le lecteur de disque, suivie par une virgule, suivie par une à huit lettres ou chiffres. Puisque nous utilisons la base B :STOCK, nous pourrons donc appeler également le format d'impression B :STOCK.

Vous pouvez vous demander comment l'ordinateur sait faire la différence entre B :STOCK (la base de données) et B :STOCK (le format d'impression). Dans un système de gestion de base de données, il y a différents « types » de fichiers et l'ordinateur a son propre système d'étiquetage pour s'y reconnaître. Le système ajoute .DBF (fichier de base de données) aux noms de fichiers tels que B :LISTTEL et B :STOCK. Dans le cas suivant, le système identifiera ce fichier comme un fichier d'impression, en ajoutant .FRM à notre nom de fichier. Nous pouvons l'appeler également B :STOCK parce que l'ordinateur sait faire la différence entre B :STOCK.DBF et B :STOCK.FRM.

Après avoir saisi le NOM DE FICHIER du document, l'ordinateur vous suggère de fournir les informations nécessaires pour préparer le document. Un document de ce type est essentiellement un format contenant des espaces vides que remplit l'ordinateur.

Lorsque le nom de fichier a été attribué à un format d'impression, il suffira de taper REPORT puis le NOM DE FICHIER, pour que l'ordinateur génère automatiquement le format d'impression. Comme nous demandons de préparer le document, nous devons répondre aux MESSAGES pour générer un format d'impression correct.

- Le deuxième message sélectionne à partir d'un menu d'options standard comprenant les choix sur la marge gauche, le nombre de lignes par page et la taille de la page. En appuyant sur la touche RETURN, vous indiquez que vous ne souhaitez pas modifier les valeurs standard du format.

- Le message « EN-TETE DE PAGE ? » (Page heading) vous demande si vous désirez un en-tête (titre) imprimé sur chaque page du document.

- Y provoquera « ENTREZ L'EN-TETE DE PAGE » à l'écran. Vous taperez alors INVENTAIRE D'UNE BOUTIQUE D'ALCOOLS.

- Le message suivant vous permet le choix entre un simple (N) ou un double (Y) espacement.

- Le message suivant vous demande si vous souhaitez des totaux.

- Si c'est le cas, le message suivant est — DESIREZ-VOUS DES SOUS-TOTAUX ?
- Si oui, le message suivant est — ENTREZ LA RUBRIQUE POUR LES SOUS-TOTAUX, c'est-à-dire sur quelle rubrique souhaitez-vous un sous-total ? Ici, nous souhaitons des sous-totaux pour chaque sorte d'ALCOOL.

- Le message suivant vous donne le choix : SUMMARY REPORT ONLY ? (Document récapitulatif seulement). Si vous répondez N, vous désirez un rapport complet (FULL REPORT). L'exemple ci-dessus est un RAPPORT COMPLET. Un « RECAPITULATIF » indiquera seulement le type d'alcool et les sous-totaux.

- Puisque le format peut être imprimé en même temps qu'il est affiché sur le terminal, on vous demande si l'imprimante doit faire un saut de page après chaque sous-total. L'ordinateur est réellement en train de vous demander si vous voulez obtenir les catégories en sous-total imprimées sur une page séparée (ou un groupe de pages). Dans notre exemple, un oui (Y) aura pour effet une impression du document sur quatre pages séparées.

- Le message suivant demande EN-TETE DE SOUS-TOTAL (SUBTOTAL HEADING). Cette option n'est souhaitable que lorsqu'il est nécessaire d'avoir un en-tête ajouté au contenu de la rubrique de sous-total. Dans cet exemple, le texte supplémentaire n'est pas nécessaire donc, nous appuyons sur la touche RETURN.

- Ensuite, l'ordinateur désire qu'on lui indique combien de colonnes comportent le document et ce qu'il y a dans chaque colonne. Pour acquérir l'information nécessaire, il vous suggérera d'entrer la taille et le contenu de chaque colonne. Les colonnes sont définies une par une et de gauche à droite.

Le contenu d'une colonne peut être soit un nom de rubrique soit une expression telle que PRIX*QUANTITE. Le document contiendra le contenu de chaque rubrique ou le résultat de l'expression.

Lorsque vous avez terminé de définir toutes les colonnes que vous souhaitez, appuyez sur la touche RETURN lorsque l'ordinateur vous suggère à nouveau de saisir la taille d'une colonne. Cela termine la tâche de génération d'un document et sauvegarde le document pour un usage ultérieur.

L'opération que nous venons de voir est présentée par l'Ecran 2-10.

```
.REPORT
DONNEZ LE NOM DU FICHIER DE GENERATION D'ETAT : B:STOCK
ENTREZ LES OPTIONS, M=MARGE GAUCHE, L=LIGNES/PAGE, W=LARGEUR DE PAGE
EN-TETE DE PAGE ? (Y/N) Y
ENTREZ L'EN-TETE DE PAGE : INVENTAIRE D'UNE BOUTIQUE D'ALCOOLS
DESIREZ-VOUS UN DOUBLE INTERLIGNE ? (Y/N) N
DESIREZ-VOUS DES TOTAUX ? (Y/N) Y
DESIREZ-VOUS DES SOUS-TOTAUX DANS VOTRE RAPPORT ? (Y/N) Y
DONNEZ LE CHAMP DE SOUS-TOTAL : ALCOOL
DESIREZ-VOUS UNIQUEMENT UN RESUME DU RAPPORT ? (Y/N) N
CHANGEMENT DE PAGE APRES LES SOUS-TOTAUX ? (Y/N) N
ENTREZ LE TITRE POUR LES SOUS-TOTAUX :
COL      TAILLE,CONTENU
001      20,MARQUE
ENTREZ L'EN-TETE : MARQUE
002      10,CONT
ENTREZ L'EN-TETE : CONTENANCE
003      3,QUANTITE
ENTREZ L'EN-TETE : QTE
DESIREZ-VOUS DES SOUS-TOTAUX ? (Y/N) Y
004      6,PXRV
ENTREZ L'EN-TETE : PX REV
DESIREZ-VOUS DES TOTAUX ? (Y/N) N
005      12,PXRV*QUANTITE
ENTREZ L'EN-TETE : VALORISATION
DESIREZ-VOUS DES TOTAUX ? (Y/N) Y
006
```

Ecran 2-10

L'aboutissement de ce travail préliminaire vous permettra ensuite à tout moment d'obtenir un document imprimé chaque fois que vous taperez REPORT et fournirez B :STOCK, comme montré à la Figure 2-1. Si vous demandez .REPORT TO PRINT à la place de .REPORT, vous obtiendrez une copie imprimée.

PAGE NO.00001
15/01/86

INVENTAIRE D'UNE BOUTIQUE D'ALCOOLS

MARQUE	CONTENANCE	QTE	PX REV	VALORISATION
SCOTCH				
LONG JACK	90 cl	32	39.00	1248
LONG JACK	1 l	7	43.00	301
LONG JACK	70 cl	88	29.80	2622.40
SOUS-TOTAL		127		4171.40
VODKA				
PETROVNA	1 l	35	23.78	823.30
PETROVNA	1 l	9	27.95	251.55
PETROVNA	90 cl	75	21.49	1611.45
SOUS-TOTAL		119		2686.30
WHISKEY				
4 ROSES	1 l 1/2	32	25.11	803.52
4 ROSES	2 l	44	31.98	1407.12
4 ROSES	90 cl	19	18.50	351.50
NEW SOUTH	1/2 l	4	12.80	51.20
SOUS-TOTAL		99		2613.04

BOURBON				
SPECIAL	70 cl	5	41.78	208.90
SPECIAL	90 cl	22	43.50	957
SPECIAL	1 l	21	58.10	1220.10
SPECIAL	1 l 1/2	3	67.30	201.90
SPECIAL	3 l	5	80.80	336.50
SOUS-TOTAL		56		2924.40
TOTAL				12395.14

Figure 2-1.

Qu'avons-nous fait ? Comment aurions-nous dû faire ? Si nous avions préparé un inventaire d'une boutique d'alcools, nous aurions utilisé très certainement un crayon, du papier et une calculette. Maintenant, nous pouvons faire tout cela avec notre système de gestion de base de données. Les résultats de l'inventaire complet seront disponibles en quelques minutes. En éliminant l'utilisation d'une calculette, nous réduisons considérablement le risque d'erreur. Et tout cela est possible sans connaissance approfondie de l'informatique, des ordinateurs, de la programmation, ou des systèmes de base gestion de base de données. Tout ce que nous avons à faire est de suivre les mêmes étapes que nous venons d'étudier précédemment. Si vous y réfléchissez, cet exemple est réellement remarquable : tout ces résultats avec deux commandes — CREATE et REPORT.

Pour obtenir une idée différente du contenu de notre stock, nous pourrions réorganiser notre listing des éléments de l'inventaire. Il est possible de l'obtenir de façon organisée sur le critère des différentes sortes de conditionnement et de marques d'alcools. Il est tout à fait possible de trier ou d'indexer une base de données. L'INDEXATION et le TRI se ressemblent en théorie mais sont complètement différents en pratique.

INDEXER OU TRIER

Si nous trions une base de données (SORT), nous réaménageons « physiquement » la base de données.



Nous allons prendre l'exemple non-informatique du tri d'un jeu de cartes. Nous pouvons arranger les cartes de toutes sortes de manière, mais d'une seule manière à la fois. Supposons que nous arrangions les cartes en suites, et par valeur dans les suites. Dans ce cas si nous cherchons le valet de carreau, nous pouvons le retrouver facilement. Si nous reprenons les cartes et nous les arrangeons par valeur (TRIEES), nous avons une suite de cartes complètement différente, mais nous savons cette fois-ci encore où trouver le valet de carreau.

Une manière intelligente d'obtenir le même résultat est d'indexer les cartes (INDEX). Cette technique laisse l'arrangement physique de la base de données dans son ordre initial. INDEX crée une liste séparée qui décrit les emplacements des éléments que nous souhaitons obtenir. Pour illustrer cela, nous reprendrons l'exemple du jeu de cartes. Sur une feuille de papier, établissons une liste de chaque carte dans l'ordre montré par la Figure 2-2.

SUITE	CARTE	POSITION
Pique	As	
	Roi	
	Dame	
	.	
	.	
	Deux	
Coeur	As	
	.	
	Deux	
Carreau	.	
	.	
Trèfle	.	
	.	

Figure 2-2.

Nous avons maintenant une liste en ordre des 52 cartes du jeu. Reprenons le jeu, et battons-le longuement.

Prenons la première carte au-dessus du jeu. Localisons la carte dans la liste et écrivons le nombre « 1 » sous le titre « POSITION » en regard de la description de la carte (Pique...As...1). Plaçons la carte sur la table et recommençons cette opération jusqu'à ce que nous ayons épuisé toutes les cartes.

La liste représente maintenant un INDEX. Chaque carte peut être retrouvée en la recherchant dans la liste et en comptant les cartes une par une à partir de la fin du jeu. Par exemple, si le valet de carreau a le nombre 10 sur la liste, c'est la 10e carte à partir de la fin du jeu. Les nombres représentant l'emplacement des cartes sur la liste sont appelés des POINTEURS.

Nous pouvons créer une liste décrivant un arrangement différent des cartes : par valeur et par suite dans les valeurs. Exactement comme il est montré à la Figure 2-3.

CARTE	SUITE	POSITION
As	Pique	
	Coeur	
	Carreau	
	Trèfle	
Roi	Pique	
	Coeur	
	.	
	.	
	Etc ...	
Etc ..	Une autre liste de cartes	

Figure 2-3.

Si nous parcourons encore une fois notre jeu et notre liste des positions des cartes, nous aurons alors deux listes qui décrirons le même arrangement physique des cartes du jeu. Aussi longtemps que nous ne changerons pas l'ordre des cartes, nous pourrons retrouver les cartes rapidement en utilisant l'un ou l'autre des INDEX.

Une bibliothèque est un autre exemple connu d'INDEX et de POINTEURS. Si nous allons dans une bibliothèque en connaissant le livre que nous cherchons, nous pouvons le trouver de deux manières : soit en le recherchant en partant du premier rayonnage, et en examinant chaque étagère une par une jusqu'à ce que nous trouvions le livre, ou bien en utilisant le catalogue des fiches qui nous indiquera sur quelle étagère se trouve le livre. Le catalogue des fiches est un index multiple. Il existe des fiches d'INDEX pour retrouver des livres sur le critère du titre, de l'auteur et du thème. Chaque fiche contient des pointeurs, sur la base de ce que nous connaissons déjà, qui nous indiquent où se trouve le livre.

Suivant le type de base de données, nous pouvons maintenir un ou plusieurs index sur les enregistrements. Nous pourrions utiliser la base de données B :STOCK et créer un index sur la rubrique contenance des différents alcools. Dans ce cas, nous aurons un FICHIER D'INDEX sur lesquels serons stockés ces enregistrements dans

l'ordre souhaité. Dans cet exemple, nous appelons le fichier index B:CNTINDEX. Nous utiliserons d'abord notre fichier inventaire B:STOCK et nous l'indexerons ensuite suivant les contenances. Contenances, bien entendu, ne signifie rien pour l'ordinateur. L'index groupe les enregistrements dans l'ordre alphabétique des contenances de la rubrique CONT. Ce qui a pour effet de grouper toutes les contenances identiques ensemble puisqu'elles commencent par la même lettre et s'épellent de la même façon.

Cette opération et son résultat sont montrés par l'Ecran 2-11.

```
.USE B:STOCK
.INDEX ON CONT TO B:CNTINDEX
00015 ENREGISTREMENT(S) INDEXE(S)
.DISPLAY ALL

00002  SCOTCH    LONG JACK    1 l         7   43.00   48.30
00005  VODKA     PETROVNA     1 l         9   27.95   37.90
00013  BOURBON   SPECIAL      1 l        21   58.10   67.15
00010  WHISKEY   NEW SOUTH    1/2 l       4   12.80   22.20
00007  WHISKEY   4 ROSES      1 l 1/2    32   25.11   40.10
00014  BOURBON   SPECIAL      1 l 1/2     3   67.30   79.12
00004  VODKA     PETROVNA     2 l        35   23.78   33.50
00008  WHISKEY   4 ROSES      2 l        44   31.98   41.00
00015  BOURBON   SPECIAL       3 l        5   67.50   79.12
00003  SCOTCH    LONG JACK    70 cl      88   29.80   50.10
00011  BOURBON   SPECIAL      70 cl       5   41.78   52.25
00001  SCOTCH    LONG JACK    90 cl      32   39.00   45.70
00006  VODKA     PETROVNA     90 cl      75   21.49   31.30
00009  WHISKEY   4 ROSES      90 cl      19   18.50   28.50
00012  BOURBON   SPECIAL      90 cl      22   43.50   51.00
```

Ecran 2-11

Par l'utilisation d'un INDEX et d'un FICHIER D'INDEX, nous laissons la base de données physique dans l'ordre de création des enregistrements. Nous pouvons vérifier que chaque numéro d'enregistrement représente bien l'emplacement qu'il occupait précédemment.

Ce qui s'est produit très similaire à l'exemple du jeu de cartes et des listes. L'affichage est contrôlé par le fichier d'INDEX B:CNTINDEX plutôt que l'ordre de création habituel. Une des choses intéressantes de INDEX c'est que vous n'avez pas à vous préoccuper de l'ordre dans lequel sont entrés les enregistrements. L'ordinateur peut les arranger rapidement suivant la séquence ou le regroupement que vous pourriez désirer.

Dans ces deux premiers chapitres, nous avons parlé des concepts de base et du travail avec un système de gestion de base de données sur micro-ordinateur. Cette initiation théorique et cette première expérience pratique seront la base pour faire des choses plus importantes et plus performantes.

Ces systèmes utilisent la vitesse de l'ordinateur pour exécuter des tâches que nous exécutions auparavant manuellement avec l'aide de « bases de données papier ». Nous avons utilisé l'analogie de la base de données papier pour illustrer le fonctionnement d'un système de gestion de base de données et pour vous initier au vocabulaire informatique.

- Le système de gestion de base de données fournit le mécanisme permettant de stocker de l'information dans l'ordinateur, de la retrouver, la manipuler et la modifier, et enfin de préparer des documents basés sur cette information.

- Les systèmes de base de données micro-informatiques ne nécessitent pas de connaissances techniques préalables de la part de l'utilisateur.

- Ces bases de données ont beaucoup de succès puisque pour la plus grande partie, elles adaptent l'ordinateur aux besoins de l'utilisateur plutôt que forcer l'utilisateur à s'adapter à l'ordinateur. Ces systèmes deviendront peut-être la base d'une utilisation « quotidienne » de l'ordinateur.


DICTIONARY

CHAPITRE III

LES SYSTEMES MATERIEL

Dans les deux premiers chapitres, nous avons fait ce que l'on appelle habituellement un « survol de la question ». Nous avons appris qu'un micro-ordinateur et un système de gestion de base de données (SGBD) sont des choses faciles à comprendre et à utiliser. Nos exemples brefs d'explication du fonctionnement d'un système de gestion de base de données et de son utilité seront développés dans les chapitres à venir. Nous reprendrons alors ces notions pour les développer plus en détail. Et, nous verrons bien sûr de nouvelles choses.

Dans ce chapitre, nous allons vous introduire aux systèmes matériel micro-informatiques. Pour utiliser un système de base de données, vous devez au préalable disposer d'un ordinateur (malgré tout — comme vous avez pu le voir précédemment — vous serez capable de comprendre et de suivre toutes ces notions sans avoir d'ordinateur). Dans le cas où vous seriez novice de l'informatique et des ordinateurs, cette brève discussion reprendra les éléments de base des systèmes d'ordinateurs et ensuite vous indiquera les choses nécessaires pour être à même d'utiliser un système de base de données. Comme vous le verrez dans ce chapitre, ainsi que les suivants, les besoins que vous avez en gestion de données sont intimement liés aux besoins en matériel informatique.

Un système micro-informatique courant est présenté à la Figure 3-1.

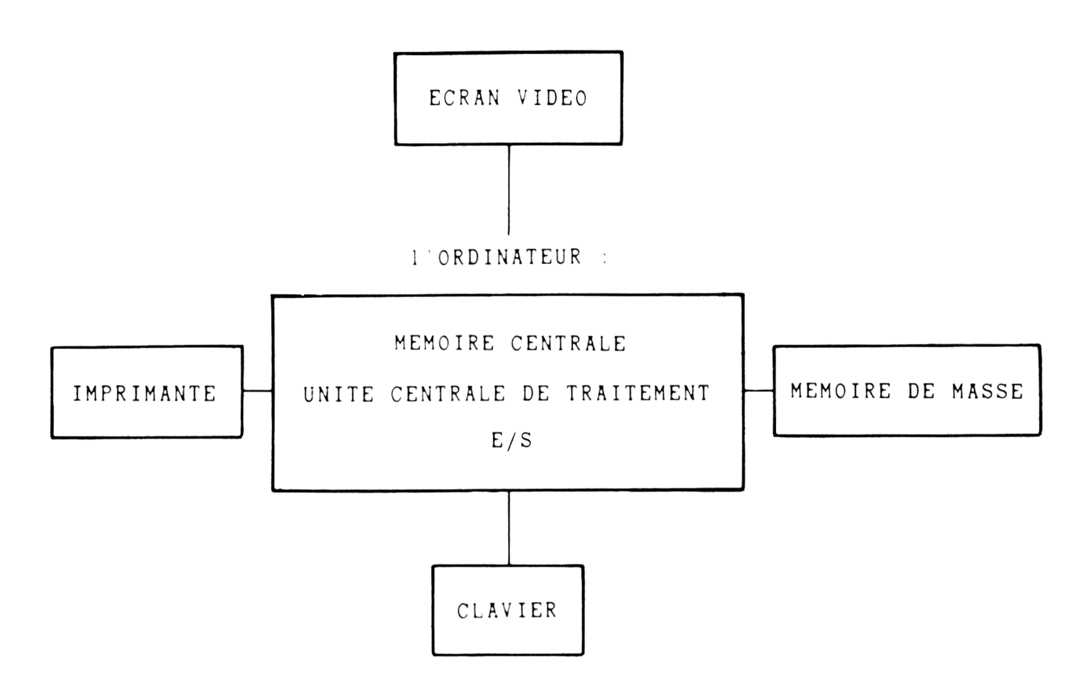


Figure 3-1. Un système de micro-informatique courant

Le diagramme précédent décrit un système quelconque d'ordinateur : ce peut être un micro-ordinateur ou un gros ordinateur. L'ordinateur lui-même représente ce qui est à l'intérieur du rectangle nommé ordinateur — l'unité centrale de traitement, la mémoire principale, et les entrées-sorties (E/S). Les quatre autres boîtes — l'écran vidéo, la mémoire de masse, l'imprimante et le clavier sont appelés des périphériques.

« *L'ORDINATEUR* » se compose de l'Unité centrale de traitement, de la Mémoire centrale et des Entrées/Sorties.

L'unité centrale de traitement est habituellement appelée CPU. C'est le circuit qui fait les « calculs ». Mais pour fonctionner, il doit disposer d'une MEMOIRE CENTRALE. La mémoire centrale est utilisée directement par le CPU.

Les publicités pour micro-ordinateurs se présentent généralement comme ceci :

« ACHETEZ UN banana DE 24 K »



Le mot 24 K représente la capacité de mémoire centrale qui est exprimée en OCTETS (bytes). Un OCTET est la capacité de mémoire nécessaire pour stocker un caractère de machine à écrire tel que « m » ou « $ ». 24 K signifie habituellement 24.000. Dans la mémoire d'ordinateur, cela signifie réellement 24.576. Un système d'ordinateur de 24 K a 24.576 OCTETS de mémoire centrale.

La mémoire centrale est relativement coûteuse lorsqu'on la compare à la mémoire de masse. Lorsque vous achetez un ordinateur, il vous faudra choisir la capacité de mémoire centrale avec beaucoup de soin. Le système de base de données que vous choisirez nécessitera un minimum de capacité de mémoire centrale. *Vous aurez besoin d'acquérir cette capacité de mémoire centrale pour votre système d'ordinateur.*

E/S s'occupe des entrées/sorties. C'est le seul moyen pour connecter l'ordinateur au monde extérieur. Dans ce cas, le monde extérieur est constitué de périphériques. Vous pourriez raisonnablement penser que les E/S sont des accessoires standard comme les roues d'une voiture. Ce n'est pas toujours vrai : les E/S d'ordinateur doivent souvent s'acquérir séparément.

Pour les micro-ordinateurs cependant, le prix est une considération relative. Aucun des composants d'un système de micro-informatique n'est très coûteux.

Les périphériques

Les ordinateurs ne nécessitent pas toujours des circuits périphériques. Cependant, pour la plupart des usages de base de données, tous les périphériques montrés par la Figure 3-1 sont nécessaires. Il s'agit de la mémoire centrale, la mémoire de masse, l'écran vidéo, et le clavier. Pour certains usages, l'imprimante est également souhaitable. Pour d'autres usages, l'imprimante peut être facultative, cependant elle sera toujours utile.

Les claviers, les moniteurs, les terminaux

Le clavier est similaire à un clavier de machine à écrire. Il est utilisé pour communiquer avec l'ordinateur. L'ordinateur communique avec vous par le moyen de l'écran vidéo. Le clavier et l'écran vidéo peuvent être des éléments séparés. Lorsqu'ils sont séparés, la vidéo est fournie par un élément ressemblant à un téléviseur appelé moniteur. Le clavier et l'écran vidéo sont souvent incorporés dans un seul et même élément. On l'appelle souvent un terminal d'ordinateur, un terminal vidéo, ou quelquefois simplement CRT.

Que vous ayez un terminal ou un clavier et un moniteur dépend généralement de la marque du système micro-ordinateur que vous possédez. Dans ce livre, nous nous référerons souvent au mot terminal pour plus de facilité. Il n'y a pas de différence pratique entre un terminal vidéo et un clavier avec un moniteur séparé. Les écrans vidéo courants peuvent afficher 24 lignes de 80 caractères en une seule fois. Quelques-uns un peu moins, d'autres un peu plus. Le clavier et l'écran vidéo sont extrêmement importants d'un point de vue subjectif : ces périphériques sont des moyens d'échange avec l'ordinateur. Vous « parlerez » à l'ordinateur au moyen du clavier ; l'ordinateur vous « parlera » au moyen de l'écran vidéo.

La mémoire de masse

La mémoire de masse est l'endroit où sont stockées la plupart des informations utilisées par un système micro-ordinateur. C'est un moyen économique pour augmenter la capacité de la mémoire centrale. La mémoire de masse est réellement moins coûteuse que la mémoire centrale. Elle est également plus lente (plus de mille fois). Lorsque l'on éteint l'ordinateur, la mémoire centrale s'efface tandis que la mémoire de masse ne s'efface pas. La mémoire de masse peut être modifiée volontairement.

La mémoire de masse est utilisée pour stocker les informations que vous voulez sauvegarder, de la même façon qu'une bande de magnétophone est utilisée pour sauvegarder les « souvenirs » de sons. Les systèmes de stockage de masse sont fabriqués de telle façon que l'ordinateur puisse facilement retrouver l'information qu'il désire. Il y a deux sortes de stockage de masse couramment utilisées par les micro-ordinateurs : la bande et le disque.

Les systèmes à bande emploient habituellement des cassettes exactement identiques à celles utilisées pour le stockage audio. Ces systèmes sont lents et mis à part quelques cas exceptionnels, ne sont pas indiqués pour être utilisés avec des systèmes de base de données.

Les systèmes à disques, appelés souvent périphériques de stockage à accès direct, sont des sortes d'« électrophones magnétiques ». L'information est stockée d'une manière magnétique sur les pistes d'un disque en rotation qui ressemble à un micro-sillon. Un électrophone à bras tangentiel, pour ceux qui connaissent, est un appareil qui lui ressemble (le lecteur de disque informatique a cependant une tête magnétique comme celle utilisée sur les magnétophones). Cette tête est utilisée pour « lire » et « écrire » sur le disque. L'information se trouve sur des « pistes » magnétiques qui sont, contrairement au microsillon invisibles à l'œil nu.

Un LECTEUR DE DISQUE est un mécanisme qui fait tourner le disque et transfère les informations. L'ordinateur « connaît » l'emplacement de chaque partie d'information sur chaque lecteur de disque. Il « connaît » cet emplacement en lisant le « catalogue » du disque de chaque disque. Le catalogue du disque ressemble à la liste des morceaux de musique d'un 33 tours : il vous dit sur quelle face du disque se trouve tel morceau de musique.

Le lecteur de disque doit être contrôlé par un contrôleur de disque qui se connecte aux E/S. Sous la commande du CPU, le contrôleur de disque dirige le mouvement de chaque tête de lecture/écriture de chaque lecteur de disque sur l'emplacement

correct du disque magnétique. Le contrôleur de disque établit une circulation d'informations entre le CPU et le disque. Il tient également informé le CPU sur le bon déroulement des opérations disque.

Pour les micro-ordinateurs, il y a deux sortes de systèmes à disque couramment utilisés : les « systèmes à disque souple » et les « systèmes à disque dur ».

Les systèmes à disque souple

Les disques souples sont souvent appelés des « disquettes » ou « floppies ». Elles ont été développées par IBM vers les années 70 pour être utilisées sur le système d'ordinateur IBM 370. C'est le support de stockage de masse les plus couramment utilisé pour les systèmes micro-informatiques. Le coût d'un disque souple est à peu près le même qu'une cassette de magnétophone.

Le prix d'un lecteur de disque souple est nettement plus élevé que le prix d'un magnétophone à cassette. Les performances d'un système à disque souple sont nettement plus importantes qu'une cassette. La performance d'un système à disque dur est nettement plus grande qu'un système à disque souple. La plupart des systèmes micro-informatiques pour les applications professionnelles possèderont un ou plusieurs lecteurs de disque souple.

Les systèmes micro-informatiques sont généralement équipés de un à quatre lecteurs de disque. Les systèmes capables de travailler avec un système de gestion de base de données auront au moins un lecteur de disque souple. Le disque souple est un moyen pratique d'introduction de nouveaux logiciels vers l'ordinateur et également pour transporter l'information ou le logiciel vers d'autres micro-ordinateurs.

Le disque souple est un film de mylar, recouvert d'oxyde, coupé à la forme d'un disque, ressemblant à un disque 45 tours d'électrophone. Ce disque est emballé dans une pochette carrée de plastique ou de papier, le trou au centre permet au lecteur de faire tourner le disque. Une ouverture sur un des côtés du carré permet à la tête de lecture/écriture d'avoir accès au disque. Un autre trou dans l'enveloppe du disque sert à positionner le disque. Lorsque le disque souple est inséré dans le lecteur de disque, l'enveloppe devient fixe tandis que le disque de mylar tourne à l'intérieur. Il est nécessaire de toujours garder propre la feuille de mylar. Le film de mylar ne doit pas être touché avec les doigts. Le fait de toucher le film ou de l'exposer à la graisse ou à la poussière peut conduire à de graves dommages pour vos données.

Il n'y a pas de standard universel pour les systèmes à disque souple. Ceci peut induire quelques problèmes lorsque l'on échange des disques souples entre les systèmes. Pour pouvoir échanger des disques souples entre les systèmes, les lecteurs de disque doivent être compatibles.

C'est également une question de taille. Les disques souples se présentent sous 5 pouces (appelés minifloppies) et en version 8 pouces. On les trouve en simple face (les données sont présentes sur une seule face du disque) ou en double face. La densité de stockage de l'information est soit de la simple densité, soit de la double densité (deux fois plus qu'en simple densité). Il y a seulement deux types de densité.

La capacité nominale de l'information que l'on peut stocker sur chacun des disques souples, suivant les types, est montrée par la Figure 3-1. Ces valeurs nous fournissent un guide général des capacités de stockage des disquettes courantes. Compte tenu de l'évolution de technologies des constructeurs de micro-ordinateurs et de lecteurs de disque, ces valeurs peuvent varier de façon importante. Ce qui n'est évidemment pas souhaitable, puisque ces variations vont à l'encontre de la compatibilité entre les systèmes.

	5 pouces	8 pouces
Simple face, simple densité	70 000	240 000
Simple face, double densité	140 000	480 000
Double face, double densité	280 000	960 000

Capacité de stockage des disques souples exprimées en octets

Tableau 3-1.

La plupart des disques simple densité 8 pouces sont enregistrés au format IBM 3740. Ce format est utilisé par le lecteur de disque pour enregistrer les données. Il est utilisé couramment par IBM pour le « 3740 KDDES ».

Chaque disque « 3740 » contient 77 pistes et se divise en 26 secteurs. Un secteur a une forme de part de gateau comme le montre la Figure 3-2. La piste la plus extérieure est la piste 0 et la piste la plus intérieure est la piste 76. Le nombre d'octets stockés sur un secteur ou une piste est généralement indépendant de la position de la piste. Ce qui veut dire que si 100 octets sont stockés sur chaque secteur de la piste 0, alors 100 octets seront stockés dans chaque secteur de la piste 76. Cet arrangement particulier permet un taux de transfert des données du disque indépendant de la piste, les pistes les plus extérieures se déplacent plus vite (centimètres par seconde) que les pistes intérieures.

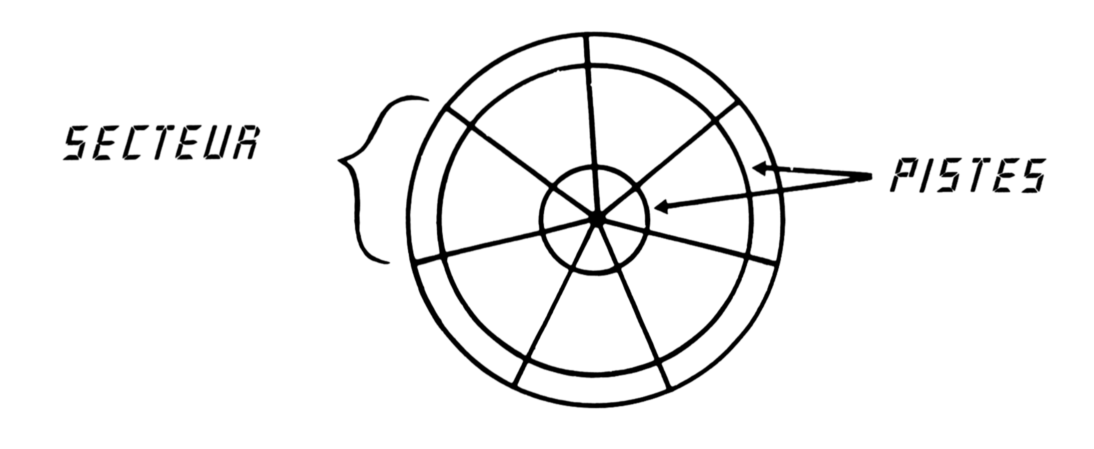


Figure 3-2.

Malheureusement, la situation de compatibilité n'est pas réalisée pour le format 5 pouces. Pour les petits disques, il n'y a pas encore de format dominant d'enregistrement. Ce qui signifie qu'il y a une volonté d'incompatibilité entre les systèmes micro-ordinateurs différents équipés de disques 5 pouces. Il est donc souhaitable de tester la compatibilité entre les systèmes avant de prévoir un échange d'informations avec des disques souples.

Les systèmes à disque dur

Bien que les disques souples soient très répandus avec les micro-ordinateurs, les disques « durs » sont réellement disponibles et opérationnels. Le disque dur veut dire simplement un système micro-informatique qui n'utilise pas de disque souple. La différence physique consiste en la rigidité du disque. Il est fabriqué à partir d'un plateau de métal usiné. Il y a des disques durs qui ne sont pas amovibles. On les appelle des « disques Winchester ».

La différence pratique c'est que les disques durs offrent des performances nettement plus importantes que les disques souples. Ils sont également beaucoup plus chers.

La capacité de stockage d'un système à disque dur se situe au-dessus du million d'octets. Les capacités de stockage de 5 à 40 millions d'octets sont des choses courantes.

Les caractéristiques de performance des systèmes à disque que vous pouvez retrouver dans la publicité sont : la vitesse piste à piste, la vitesse d'accès moyenne, et la vitesse de transfert des données. Ce sont les mesures de performance de ce périphérique particulier.

Piste à piste signifie le temps mis par le lecteur de disque pour déplacer la tête d'une piste à une autre piste adjacente. Les valeurs exprimées dans la documentation technique sont généralement de quelques millisecondes. Une milliseconde est 1/1000 d'une seconde.

La vitesse d'accès moyenne représente le temps moyen requis pour repositionner la tête. Plus précisément, c'est le temps nécessaire pour déplacer la tête de la piste 1 à la dernière piste, divisé par 2. Il est normalement de quelques millisecondes pour les systèmes à disque dur, et on ne le trouve presque jamais dans les publicités pour les lecteurs de disque souple (100 à 500 millisecondes). C'est une caractéristique technique que vous ignoriez sans doute et qui ne doit pas être perdue de vue.

Une autre caractéristique commerciale que vous pouvez ignorer : la vitesse de transfert des données. Les vitesses de transfert de données sont normalement exprimées en centaines de milliers jusqu'au million d'octets par seconde.

Les disques 8 pouces à double densité sont habituellement enregistrés au format IBM 34. Le fait d'enregistrer les données sous un format compatible ne garantit pas toujours que les disques puissent être échangés entre les micro-ordinateurs, mais

c'est déjà un progrès. Il existe un grand nombre de disques 8 pouces qui peuvent être échangés entre des systèmes micro-ordinateurs différents.

Les deux dernières caractéristiques techniques s'appliquent au lecteur de disque des gros systèmes. La question qui se pose est leur emploi dans un système micro-ordinateur. Il sera plus intéressant pour vous de savoir si l'appareil de lecture de disque est compatible avec votre ordinateur — si c'est le cas, qu'existe-t-il pour connecter l'appareil et le faire fonctionner correctement avec votre ordinateur ? Pour information, on appelle INTEGRATION la connexion de l'appareil et sa mise en fonction correcte.

Pour le sujet qui nous occupe, la mémoire de masse a une importance particulière. Les bases de données sur micro-ordinateurs sont habituellement stockées sur disque de l'un ou l'autre type. Aucune démarche de gestion de base de données ne fonctionnera si vous n'avez pas le stockage disque adéquat. La plupart des micro-ordinateurs couramment disponibles ne gèrent qu'une très petite partie de la base de données en mémoire centrale. Les bases de données très importantes requièrent habituellement le disque dur. Les bases de données de taille moyenne peuvent être contenues sur des disques souples de l'un ou l'autre type.

Même les bases de données importantes qui sont contenues sur des disques durs peuvent être transférées sur des disques souples (« back up »). Le back up signifie la copie intégrale des données. Cette copie supplémentaire sert à se prémunir contre les pannes de disque et fournit un moyen économique de transfert d'une base de données d'un micro-ordinateur à l'autre.

L'imprimante

Une imprimante est toujours utile même si elle n'est pas absolument obligatoire. Le choix particulier de l'imprimante n'entre pas dans la discussion sur les bases de données de façon prépondérante. Il y a deux sortes d'imprimantes de base : à aiguille et à marguerite.

- L'imprimante à marguerite fournit des caractères de la même qualité qu'une machine à écrire. Cette imprimante convient généralement lorsque l'on désire une qualité lettre.
- L'imprimante à aiguille utilise un certain nombre de petites aiguilles pour former le caractère. On l'appelle aussi matricielle.

Il y a des variations considérables dans la qualité d'impression des imprimantes à aiguilles. Les imprimantes à aiguilles sont généralement moins chères et d'une qualité d'impression inférieure que les imprimantes à marguerite. Elles sont par contre beaucoup plus rapides. Les vitesses d'impression courantes pour les imprimantes à marguerite sont de 40 à 50 caractères par seconde. Les vitesses d'impression pour les imprimantes à aiguilles sont souvent supérieures à 100 caractères par seconde. L'impression à aiguilles ne permet pas toujours de tirer des photocopies.

Le système d'exploitation

Lorsque la plupart des micro-ordinateurs sont allumés, la mémoire centrale est vierge : la machine n'a aucun objectif et ne sait pas faire grand chose. Pour rendre l'ordinateur opérationnel, vous avez besoin d'un SYSTEME D'EXPLOITATION. Ce système permet à votre système de gestion de base de données de contrôler l'ordinateur ainsi qu'à vous-même.

La plupart des systèmes micro-ordinateurs qui peuvent recevoir des systèmes de gestion de base de données utilisent un système d'exploitation disque (DOS). Lorsqu'on allume l'ordinateur, le système d'exploitation est chargé en mémoire centrale à partir du disque.

Cela peut se faire automatiquement, mais il est parfois nécessaire d'appuyer sur un bouton de l'ordinateur. L'opération de chargement du système d'exploitation s'appelle BOOTER le système. Ne soyez pas inquiet, il ne vous sera pas nécessaire d'apprendre beaucoup de choses sur le système d'exploitation — maintenant ou plus tard — pour utiliser un système de base de données.

Ce livre sous-entend l'utilisation du système d'exploitation CP/M Plus, le système utilisé par les micro-ordinateurs Amstrad. Les seules choses qui peuvent changer avec d'autres systèmes d'exploitation sont les règles pour adresser les lecteurs de disque et/ou nommer les fichiers.

Encore quelques mots sur les questions de matériel

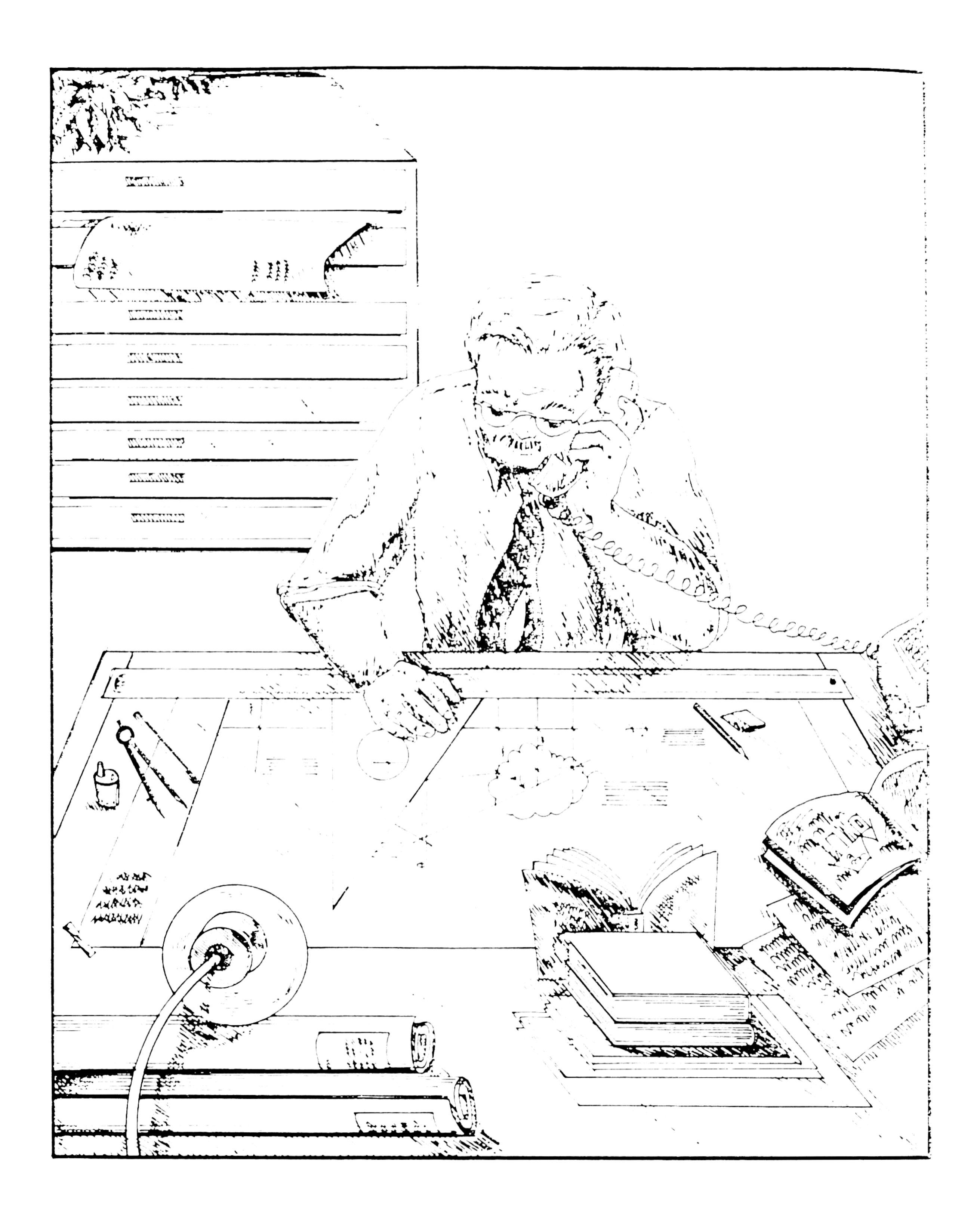
Pour pouvoir commencer correctement, il vous est nécessaire d'avoir un matériel informatique adéquat.

Le système lecteur de disque est l'élément matériel le plus critique pour un système de gestion de base de données sur ordinateur. Ce système devra avoir suffisamment de capacité de stockage pour ranger toute la base de données sur un simple disque. (Au Chapitre 3, nous verrons comment déterminer la taille de notre base de données en OCTETS). Le CPU ne pourra utiliser que l'information contenue sur les lecteurs de disque qui lui seront connectés. Ces données sont dites ON-LINE. Si vous avez des lecteurs de disque souple, seuls les disques souples qui seront réellement insérés dans les lecteurs de disque sont ON-LINE. Si vous souhaitez acquérir un ordinateur et l'utiliser dans une application de base de données, vous devrez évaluer consciencieusement votre capacité disque.

Si vous n'avez pas encore choisi d'ordinateur, il serait raisonnable pour vous et votre carnet de chèques de suivre la procédure suivante : choisir le système de base de données, déterminer vos besoins de stockage, choisir un système d'ordinateur qui (1) est compatible avec votre système logiciel de base de données et (2) qui supportera vos besoins de stockage et offrira un moyen raisonnable pour ajouter des informations supplémentaires si cela devient nécessaire.

DEUXIEME PARTIE

Dans la deuxième partie, nous allons développer un exposé encore plus clair sur la manière de concevoir et d'utiliser une base de données. Nous verrons d'abord la facilité et l'importance d'une bonne conception puis nous passerons à la construction, la modification, la maintenance et, finalement l'utilisation de bases de données conséquentes. Nous travaillerons sur les concepts et les thèmes déjà rencontrés dans la première partie.



CHAPITRE IV

CONCEPTION DE VOTRE BASE DE DONNEES

L'élaboration d'un projet est souvent ressenti comme une chose inutile, particulièrement pour le débutant. Les autres ont quelquefois ce sentiment. Malheureusement, si vous ne préparez pas suffisamment un projet, vous risquez de ne pas être satisfait du résultat et vous pourriez évidemment avoir à tout refaire une deuxième fois.

Reprenons l'exemple d'une base de données papier en utilisant une machine à écrire. Si la secrétaire ne prévoit pas correctement la place des colonnes, la base de données dépassera à coup sûr la marge droite du papier. Ce n'est pas une catastrophe, mais c'est désagréable et il sera nécessaire de recommencer.

La même chose s'applique avec votre base de données d'ordinateur. Elle ne va pas dépasser de la feuille de papier, mais si elle n'est pas correctement conçue, vous serez obligé de revenir en arrière et de faire un travail supplémentaire pour obtenir le résultat souhaité. En fait, la bonne démarche pour planifier votre base de données, c'est de la revoir globalement deux ou trois fois. Ce processus agit par AMELIORATIONS SUCCESSIVES. Un des avantages du travail avec une base de données ordinateur (contrairement à une base de données papier), c'est qu'il est possible de procéder à des modifications importantes sur la base de données sans avoir à ressaisir les données. L'ordinateur, dans bien des cas, peut récupérer la plupart — si ce n'est toutes — les données stockées dans la base avant les dernières modifications.

La première étape dans la conception de votre base de données est de savoir ce que vous désirez en faire. Lorsque vous savez ce que vous voulez mettre à l'intérieur, l'étape suivante consistera à élaborer une liste des éléments qui y seront inclus. Ne soyez pas perfectionniste. A peu près toutes les imperfections seront facilement arrangées ou supprimées.

Comme premier exemple, examinons de nouveau la base de données d'une boutique d'alcools créée au Chapitre II. Cette base de données « inventaire » va nous servir à consulter le stock et à connaître sa valeur. Lorsque nous avons créé cette base de données, nous avons simplement établi les titres des colonnes pour définir la structure de l'enregistrement. Si nous avions préparé cette base de données, nous aurions établi une liste des éléments inclus dans cette base :

Marque d'alcool
Contenance des bouteilles
Type d'alcool
Prix de revient
Prix de vente
Quantité disponible

Cette liste décrit ce qui se trouve réellement dans l'inventaire d'une boutique d'alcool du Chapitre II. Comme nous réfléchissons à cet exemple, il y a un élément supplémentaire qui pourrait y être ajouté :

Emplacement de l'article

Cet oubli est simple à corriger. Nous reverrons le processus à suivre pas à pas, et vous verrez comment il est facile de faire cette modification. Vous aurez alors l'esprit en paix.

Nous possédons déjà une base de données, B :STOCK, qui contient un certain nombre de données. Que faire ? Nous voulons ajouter une rubrique et nous ne voulons pas perdre les données qui ont déjà été saisies.

AUCUN PROBLEME ? Nous pouvons résoudre cette difficulté à partir du clavier en quelques minutes, sans perdre aucune donnée. Et même plus, il y a deux façons d'y parvenir.

Solution 1 : Créer un nouveau fichier

Une solution envisageable serait de créer un nouveau fichier (nous l'appellerons B :NVXFICH) ayant la même structure (noms de rubriques, types de rubriques et tailles) que B :STOCK, sur lequel on ajoutera une nouvelle rubrique pour tenir compte de l'emplacement des articles dans le magasin. Nous créons B :NVXFICH en utilisant la commande dBASE II CREATE. Une fois le nouveau fichier créé, on ajoutera toutes les informations stockées dans B :STOCK au nouveau fichier.

L'opération de création est montrée par l'Ecran 4-1. Notez que nous avons choisi de placer la nouvelle rubrique EMPLACMT au milieu de l'enregistrement. Nous l'avons fait pour illustrer comment le système de gestion de base de données manipule les rubriques — la position de la rubrique dans l'enregistrement n'est pas primordiale.

```
 .CREATE
 ENTREZ LE NOM DE FICHIER : B:NVXFICH
 ENTREZ LA STRUCTURE DE L'ENREGISTREMENT COMME SUIT :
 CHAMP      NOM,TYPE,DIM,DECIMALE(S)
 001        ALCOOL,C,10
 002        MARQUE,C,20
 003        CONT,C,7
 004        QUANTITE,N,3
 005        EMPLACMT,C,10
 006        PXRV,N,6,2
 007        PXVT,N,6,2
 008

VOULEZ-VOUS COMMENCER LA SAISIE (Y/N) ? Y
```

Ecran 4-1

Notre décision, dans cet exemple, sera de ne pas entrer de données, donc N. Le fait de répondre non laisse de côté la base de données qui vient d'être créée. Pour travailler de nouveau avec celle-ci, nous devons utiliser la commande USE. Toutes les données stockées dans le fichier B:STOCK peuvent être ajoutées au fichier B:NVXFICH par les commandes présentées à la Figure 4-2.

```
.USE B:NVXFICH
.APPEND FROM B:STOCK
00015 ENREGISTREMENT(S) TRANSFERE(S)
```

Ecran 4-2

Dans le très court exemple ci-dessus, une nouvelle base de données a été créée par le processus de la commande CREATE. Toutes les informations stockées dans le fichier de la base de données B :STOCK sont ajoutées au nouveau fichier (B :NVXFICH) en utilisant la commande dBASE II APPEND. Toutes les informations apparaissent dans leur propre rubrique, même après l'ajout d'une nouvelle rubrique — EMPLACMT — en plein milieu de notre base de données.

Nous disposons maintenant de deux fichiers, B :NVXFICH et B :STOCK, qui contiennent chacun les mêmes informations. B :NVXFICH contient la rubrique additionnelle : EMPLACMT. Nous souhaitons appeler ce fichier B :STOCK. B :STOCK ne nous est plus d'aucune utilité. Nous pouvons donc nous en débarrasser et renommer B :NVXFICH comme B :STOCK avec les commandes présentées par l'Ecran 4-3.

```
.DELETE FILE B:STOCK
LE FICHIER A ETE EFFACE
.RENAME B:NVXFICH TO B:STOCK
```

Ecran 4-3

Solution 2 : modification de la structure

dBASE II vous permet de modifier la structure d'une base de données de manière similaire à la commande EDIT. Nous utiliserons alors la commande MODIFY STRUCTURE. Malheureusement, lorsque nous « modifions » la structure de cette manière, toutes les données sont détruites. Pour faire face à cette situation, nous pouvons effectuer la « modification » en faisant une copie de la structure et en « modifiant » ensuite notre copie de structure. La base de données originale est toujours intacte. Nous procéderons avec les commandes comme présenté par l'Ecran 4-4.

```
.USE B:STOCK
.COPY STRUCTURE TO B:NVXFICH
.USE B:NVXFICH
.MODIFY STRUCTURE
LA COMMANDE MODIFY DETRUIRA TOUTES VOS DONNEES... (Y/N) ? Y
```

Ecran 4-4

Nous pouvons maintenant, en toute sécurité, modifier la structure de la base B :NVXFICH sans affecter la base de données originale B :STOCK. L'Ecran 4-5 est la réplique exacte de notre base de données originale B :STOCK. Même si elle ressemble étrangement à B :STOCK, c'est réellement le fichier dupliqué, B :NVXFICH. B :STOCK existe encore, il est mis en réserve pendant l'opération de modification de la structure.

Dans cet exemple, nous insérerons une nouvelle rubrique entre QUANTITE et PXRV. Pour y parvenir, nous appuyons sur la touche RETURN quatre fois de suite. Le curseur se positionnera au début du nom de rubrique PXRV. Ensuite,

- Appuyez en même temps sur la touche « Ctrl » et sur la touche « N ».

Il y aura insertion d'un espace à la rubrique 5. Ensuite,

- Tapez le nouveau nom, le type et la taille de la rubrique.

```
                NOM          TYPE     LONG     DEC
CHAMP 01 : ALCOOL             C        010     000      :
CHAMP 02 : MARQUE             C        020     000      :
CHAMP 03 : CONT               C        007     000      :
CHAMP 04 : QUANTITE           N        003     000      :
CHAMP 05 : PXRV               N        006     002      :
CHAMP 06 : PXVT               N        006     002      :
CHAMP 07 :                                              :
CHAMP 08 :                                              :
CHAMP 09 :                                              :
CHAMP 10 :                                              :
CHAMP 11 :                                              :
CHAMP 12 :                                              :
CHAMP 13 :                                              :
CHAMP 14 :                                              :
CHAMP 15 :                                              :
CHAMP 16 :                                              :
```

Ecran 4-5

L'affichage ressemble à l'Ecran 4-6 ci-après.

	NOM	TYPE	LONG	DEC	
CHAMP 01 :	ALCOOL	C	010	000	:
CHAMP 02 :	MARQUE	C	020	000	:
CHAMP 03 :	CONT	C	007	000	:
CHAMP 04 :	QUANTITE	N	003	000	:
CHAMP 05 :	EMPLACMT	C	001	000	:
CHAMP 06 :	PXRV	N	006	002	:
CHAMP 07 :	PXVT	N	006	002	:
CHAMP 08 :					:
CHAMP 09 :					:
CHAMP 10 :					:
CHAMP 11 :					:
CHAMP 12 :					:
CHAMP 13 :					:
CHAMP 14 :					:
CHAMP 15 :					:
CHAMP 16 :					:

Ecran 4-6

Appuyez sur Ctrl W. La base de données B :NVXFICH possède maintenant la nouvelle rubrique que nous voulions lui adjoindre. Les données existantes de B :STOCK peuvent être ajoutées et le fichier peut être renommé exactement comme dans l'exemple précédent (Ecran 4-2 et Ecran 4-3). Pour ajouter les données de B :STOCK :

```
.LIST
```

Maintenant, nous avons à la fois la nouvelle et l'ancienne version de notre base de données. Une bonne habitude consiste à examiner notre nouvelle base de données pour s'assurer que tout est correct, et d'effacer ensuite notre ancienne version.

```
.APPEND FROM B:STOCK
00015 ENREGISTREMENT(S) AJOUTE(S)
```

Lorsque la réponse de l'ordinateur à notre demande de « listage » apparaît sur l'écran, nous observons un espace vierge supplémentaire entre les colonnes QUANTITE et PXRV. Celui-ci contiendra les valeurs de notre nouvelle rubrique EMPLACMT. Si tout semble correct, nous pouvons maintenant nous débarrasser de l'ancien fichier B :STOCK.

Tapons simplement :

```
.DELETE FILE B:STOCK
LE FICHIER A ETE EFFACE
.RENAME B:NVXFICH TO B:STOCK
```

Ceci termine le processus de modification de la structure d'une base de données de la Solution 2.

C'est un peu comme un gâteau sans rien à l'intérieur. Vous pouvez y placer ce que vous voulez (CREATE) puis, par des AMELIORATIONS SUCCESSIVES, vous descendrez plus profondément dans l'application et réaliserez des choses que vous ne soupçonniez pas lorsque vous avez commencé.

Revenons maintenant un peu en arrière sur le concept de planification précédant les commandes de CREATION et d'UTILISATION. Il est temps maintenant d'en savoir plus sur les caractéristiques de fonctionnement et les limitations d'une base de données.

Si nous avions mis en place une base de données papier en nous servant d'une machine à écrire, nous aurions eu besoin de faire deux choses : attribuer un titre à chacune des colonnes et calculer l'espace nécessaire pour chacune de ces colonnes. Nous devons réfléchir à ces deux choses pour établir une base de données informatique. Le type d'information dans chaque colonne fait partie de la préparation de notre base de données.

Trois sortes de rubriques sont utilisées dans les bases de données informatiques. Ce sont :

les RUBRIQUES CARACTERES
les RUBRIQUES NUMERIQUES
les RUBRIQUES LOGIQUES.

Les RUBRIQUES CARACTERES sont les rubriques du type le plus courant. Une rubrique caractère peut contenir tout ce qui peut être imprimé par une machine à écrire. Cela comprend des lettres (minuscules et majuscules), des nombres et des symboles particuliers tels que ?, &, <, etc, ainsi que « espace ». Une rubrique caractère s'utilise habituellement pour un usage général. En réalité, nous pouvons attribuer le type caractère à chaque rubrique de la base de données. Quelques exemples typiques de rubriques caractères sont représentés par NOM, ADRESSE, et TELEPHONE dans notre exemple de catalogue de téléphones au Chapitre I.

La taille d'une rubrique caractère représente le nombre d'espaces caractère nécessaires pour contenir la taille d'information maximum pour cette rubrique. Chaque lettre, nombre, symbole spécial et « espace » compte comme un caractère. Chaque caractère occupe un OCTET en mémoire. Chaque fois que nous faisons la relation entre la taille de la rubrique par rapport à l'espace sur une page dactylographiée, nous pouvons faire le parallèle entre cet espace et la mémoire de l'ordinateur. Le champ taille est toujours exprimé en OCTETS. Le nombre d'octets sera le même que le nombre d'espaces requis pour placer la rubrique sur une page dactylographiée.

Les RUBRIQUES NUMERIQUES ne contiennent que des nombres. On utilise uniquement de type de rubrique pour pouvoir faire des calculs. Elles peuvent contenir soit des nombres entiers (appelés integers) ou des nombres décimaux. En plus des nombres tels que 1 et 8, elles peuvent contenir des points décimaux et des signes moins (—). Un nombre négatif, tel que — 28165, occupe sept espaces (OCTETS) et dispose de 2 positions décimales. Dans presque tous les systèmes de base de données, le signe positif (+) est implicite et n'a pas besoin de figurer. Le signe moins et le point décimal occupent chacun un espace et doivent être pris en compte lorsque l'on calcule la taille de la rubrique.

Les rubriques numériques sont également « justifiées à droite » (par l'ordinateur). Pour cette raison, elles sont souvent utilisées pour des nombres qui ne seront pas repris dans des calculs mais qui devront être présentés justifiés à droite. Les rubriques caractères sont toujours « justifiées à gauche » par l'ordinateur. Des exemples de colonnes de nombres justifiées à droite ou justifiées à gauche sont représentés par la Figure 4-1.

JUSTIFIE A GAUCHE	JUSTIFIE A DROITE
1	1
10	10
100	100

Figure 4-1. Justification

Les RUBRIQUES LOGIQUES sont utilisées s'il n'y a seulement que deux possibilités pour les données. Par exemple, les factures sont réglées ou ne le sont pas. Les étudiants sont admis dans une classe ou ne le sont pas. Vous êtes en train de lire ce texte ou non. Les rubriques logiques occupent toujours un seul OCTET (espace). Les données peuvent être saisies comme T ou F (pour VRAI = TRUE ou FAUSSE = FALSE), ou alternativement comme O ou N (Oui ou Non).

Examinons les raisons qui déterminent le type de rubrique que nous choisirons. Au Chapitre I, nous avons vu que le nom de rubrique pouvait avoir de 1 à 10 caractères. Les noms de rubriques doivent être choisis de manière très significative, mais aussi courts que possible pour une raison pratique. Par exemple, supposons que nous ayons des rubriques qui contiennent les quantités en stock de chaque .mois de l'année. Si nous le souhaitons, nous pouvons avoir des noms de rubriques comme JANVIER, FEVRIER, etc. Mais JAN, FEV, etc., conviennent parfaitement.

Une autre « décision importante » s'applique aux rubriques numériques : contiendront-elles des nombres décimaux et si c'est le cas, combien de positions maximum il faut prévoir ? Cette décision est un peu anticipée lors de la planification mais nous la prendrons au moment de la création de la structure (CREATE) sur l'écran.

Nous allons travailler sur le projet de notre base de données stock que nous avons créée au Chapitre II. Son nom courant est B :STOCK, elle a été modifiée afin d'ajouter l'élément emplacement des articles. Nous utiliserons les mêmes noms, types et tailles de rubriques, que nous avons utilisés dans l'exemple. La nouvelle rubrique contient l'élément emplacement qui a été nommé EMPLACMT. Ce sera une rubrique caractère. Pour cet exemple, nous déclarerons arbitrairement une taille de 10 caractères à cette rubrique.

Le plan d'une base de données STOCK ressemble à ce qui est présenté à la Figure 4-2. Notez qu'elle ressemble à la base de données elle-même. Si vous avez plusieurs bases de données, il est judicieux d'avoir une base de données qui contienne les plans des différentes bases de données. Cette base de données de plans s'appelle un DICTIONNAIRE DE DONNEES.

DESCRIPTION DE LA RUBRIQUE	NOM DE LA RUBRIQUE	TYPE	TAILLE	DECIMALES
Type d'alcool	ALCOOL	C	10	
Marque d'alcool	MARQUE	C	20	
Contenance	CONT	C	7	
Prix de revient	PXRV	N	6	2
Prix de vente	PXVT	N	6	2
Quantité en stock	QUANTITE	N	3	
Position de l'élément	EMPLACMT	C	10	
NOMBRE TOTAL D'OCTETS			62	
NOMBRE D'ENREGISTREMENTS PREVUS			1000	

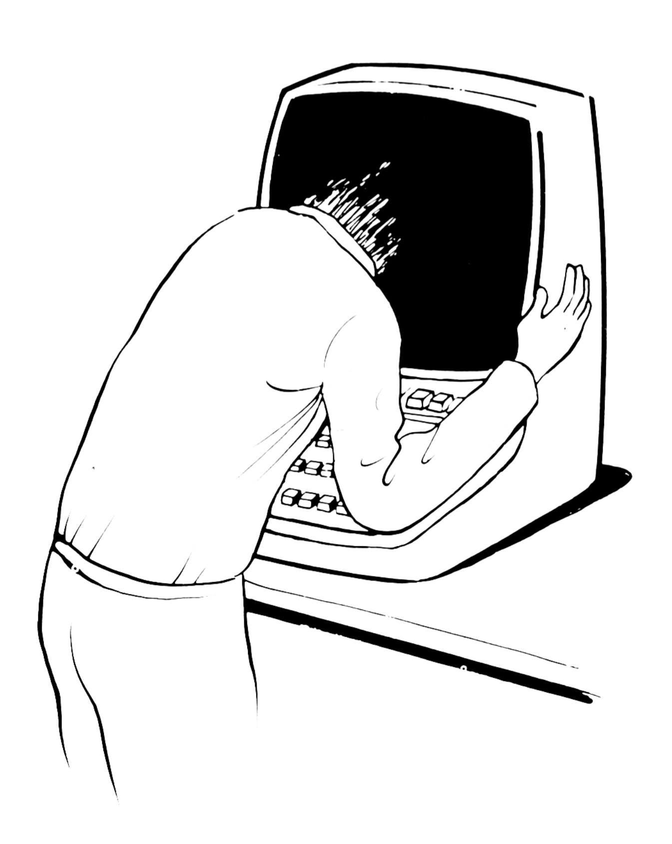
Figure 4-2. Exemple d'un plan d'une base de données pour B:STOCK

Cette base de données a sept rubriques et nécessite 62 octets de mémoire pour chaque enregistrement. Avant de vous dire — « quelle histoire, pourquoi dois-je me plier à toute cette procédure alors que je peux la faire mentalement ? » — vous devez prendre en compte que votre base de données pourrait facilement comporter de plusieurs douzaines de rubriques avec des centaines d'octets pour chaque enregistrement. Si c'est le cas, vous devez comparer votre plan avec les ressources dont vous disposez. Chaque système de base de données, de même que chaque ordinateur, a ses limites qui doivent être prises en compte si votre application devient vraiment plus importante.

Les premières limitations qui affectent un système de gestion de base de données consistent principalement en la capacité des lecteurs de disque. Par conséquent, elles peuvent être résolues soit en ajoutant des lecteurs de disque soit en utilisant des lecteurs de disque de capacité plus importante.

Il est heureux de noter que la taille d'une base de données est souvent proportionnelle aux ressources financières disponibles. Ceux qui ont des besoins de bases de données importantes ont généralement les ressources correspondantes. Les limitations du stockage de masse sont résolues par des solutions matérielles qui représentent financièrement un coût de l'ordre de cinq mille à vingt mille francs.

Les limites imposées par un système de base de données sont plus importantes que les limites dans l'espace de stockage mémoire. Ces limites peuvent vous amener à rechercher un nouveau système de base de données qui serait la panacée pour résoudre tous vos problèmes. L'alternative à cette démarche coûteuse c'est de faire fonctionner votre cerveau.



Les limitations de ressources imposées par le système de base de données se résument à ce qui suit :

Nombre de rubriques
Taille des rubriques
Nombre d'octets par enregistrement
Nombre d'enregistrements dans le fichier (base de données)

Avec dBASE II, 32 rubriques sont permises, chacune est limitée à 254 octets, et chaque enregistrement a 1 000 octets. Le nombre d'enregistrements est limité à 65 535. Ce qui veut dire que la taille maximum pour chaque fichier de base de données dBASE II est 65 535 000 octets. C'est considérable. C'est tellement énorme que vous n'atteindrez pratiquement jamais ces limites.

Pour vous donner une idée de ces caractéristiques, considérons la chose suivante. Prenons une machine à écrire standard, du papier standard de format 21 × 29,7, des marges de 2 cm, et un caractère de type Pica. La base de données considérée plus haut nécessiterait 18 670 pages une fois recopiée sur papier. Un micro-ordinateur avec un disque Winchester peut « lire » la base de données à environ 16 000 caractères par seconde. Ce qui veut dire qu'il faudrait à l'ordinateur plus d'une heure pour « lire » la base de données. Lorsque vous avez affaire à des bases de données aussi importantes, vous rencontrerez sans doute des limites soit matérielles soit dues au système d'exploitation. CP/M 2.2, le système d'exploitation le plus connu pour micro-ordinateur, limite les fichiers à 8 millions d'octets par exemple.

Un des problèmes les plus courants, lorsque l'on conçoit une base de données, survient lorsque le plan nécessite plus de rubriques qu'en offre le système de base de données. La solution est simple. Il suffit simplement de partager le plan en deux ou trois bases de données. Si c'est la cas, chaque base de données devient alors un « fichier » dans une base de données plus importante et vous pourrez vous référer à chacune de ces bases comme à un fichier. Tous ces fichiers pris ensemble constitueront la base de données.

Lorsque vous partagez votre plan en deux ou plusieurs fichiers, vous devez trouver un moyen de lier les bases de données ensemble. Le moyen d'y parvenir est d'avoir une ou plusieurs rubriques communes dans chaque fichier de base de données. Comme exemple, nous allons étudier une base de données d'une école élémentaire. Le plan de cette base de données hypothétique qui comporte 60 rubriques se présente comme à la Figure 4-3.

RUBRIQUE	DESCRIPTION DE LA RUBRIQUE	NOM DE LA RUBRIQUE	TYPE	TAILLE	DECIMALES
1	Nom de l'étudiant	NOM	C	30	
2	Classe	CLASSE	C	3	
3	Niveau	NIVEAU	C	1	
4	Nom du professeur	PROF	C	15	
5	Admis l'an dernier Y/N	ADMIS	L	1	
.	.	.	.	.	
.	.	.	.	.	
57	Adresse personnelle	ADRESSE	C	30	
58	Numéro téléph. domic.	TEL	C	8	
59	Responsable	RESNOM	C	30	
60	Téléphone d'urgence	URGENCE	C	8	
NOMBRE TOTAL D'OCTETS					341
NOMBRE D'ENREGISTREMENTS PREVUS					600

Figure 4-3. Plan d'une base de données pour une école élémentaire

Voilà un bon exemple de base de données très importante. Elle nécessite un stockage de masse d'environ 200 000 octets (caractères). On peut dire que 90 % de toutes les bases de données comportent moins de 100 000 octets. Cet exemple donne une idée des nécessités professionnelles que l'on rencontre couramment.

VOUS DEVEZ CONNAITRE VOTRE APPLICATION
AVANT D'ACHETER VOTRE MATERIEL ET VOTRE LOGICIEL

Pour cette application, la configuration matérielle minimum doit comporter deux lecteurs de disque ou au moins un lecteur capable de stocker environ 500 000 octets. Vous pourriez faire avec moins mais vous seriez contraint d'abandonner certaines opérations souhaitées. Par exemple, si vous voulez ajouter une rubrique à une base de données, vous aurez besoin de 400 000 octets pour ranger les données jusqu'au moment où vous aurez récupéré tous les enregistrements d'une base de données dans l'autre.

La seule limite logicielle (pour dBASE II), dans cet exemple, se trouve être le nombre de rubriques nécessaires : il y a plus de rubriques qu'un fichier de base de données

peut en supporter (32). La solution sera de partager la base en deux ou plusieurs fichiers. Lorsque les bases de données sont séparées et occupent deux ou plusieurs fichiers, nous devons les relier ensemble. Pour celà, on utilise un élément « commun » : une donnée significative pour l'ensemble de la base, tel que le nom de l'élève.

Ce n'est pas très compliqué. Si l'on transposait sur papier, chaque fichier élève serait placé sur plus d'une feuille de papier. Sur un fichier papier, le nom des élèves apparaîtrait sur chacune des feuilles. Dans notre base de données, chaque fichier possède une rubrique NOM qui contient le nom de l'élève. Le nombre total de rubriques est maintenant 61 et le nombre total de caractères est 371. Il y a 61 rubriques parce que nous avons ajouté « NOM » de nouveau dans la rubrique 33 pour identifier les informations des rubriques 34 à 60. On installe donc un élément commun pour pouvoir partager en deux la base de données.

Un problème se posera s'il y a dans l'école deux élèves ayant le même nom. Ce n'est pas très courant. Nous dépasserons ce problème en créant trois rubriques — NOM, CLASSE et NIVEAU — communes aux deux fichiers. Le plan nécessite maintenant 63 rubriques de 375 octets par « enregistrement ». Avec dBASE II, ce plan peut s'appliquer avec deux fichiers formant l'ensemble de la base de données.

Une autre solution consisterait à affecter un numéro d'identification à chaque étudiant. Ceci a l'avantage de prendre moins d'octets que la première solution et nécessite seulement deux rubriques supplémentaires au lieu de trois. Le seul point négatif vient du fait que l'utilisateur doit fournir un numéro d'identification unique pour chaque élève.

Examinons les rubriques CLASSE, NIVEAU et PROF. Dans beaucoup d'endroits cette information sera redondante. Dans une école élémentaire, M. Jones sera affecté uniquement à la classe 201 de niveau CM2. Dans ce cas, il serait plus raisonnable et efficace d'établir un troisième fichier de base de données qui contiendrait des informations sur les professeurs. Comme l'école dispose certainement d'un fichier du personnel, on peut donc éliminer une rubrique du fichier élèves. Dans ce cas particulier, on réalise une économie de 9 000 octets de mémoire (600 enregistrements par 15 octets), plus l'économie de frappe de 600 noms, et la modification entière lors du remplacement d'un professeur.

Cette base de données scolaire a maintenant trois fichiers. Ces fichiers sont reliés entre eux. C'est finalement la vraie définition d'un système de base de données RELATIONNEL. Les fichiers peuvent être reliés les uns aux autres pour minimiser la duplication d'informations. En fait, le terme technique d'un fichier de base de

données est le mot RELATION. Il est généralement souhaitable de regrouper l'information en une simple relation. La raison est la suivante : il est plus facile de travailler avec une base de données qu'avec deux ou plus.

Dans notre exemple, les trois fichiers de base de données sont reliés comme le montre la Figure 4-4. Celle-ci illustre la façon dont les fichiers peuvent être mis en relation.

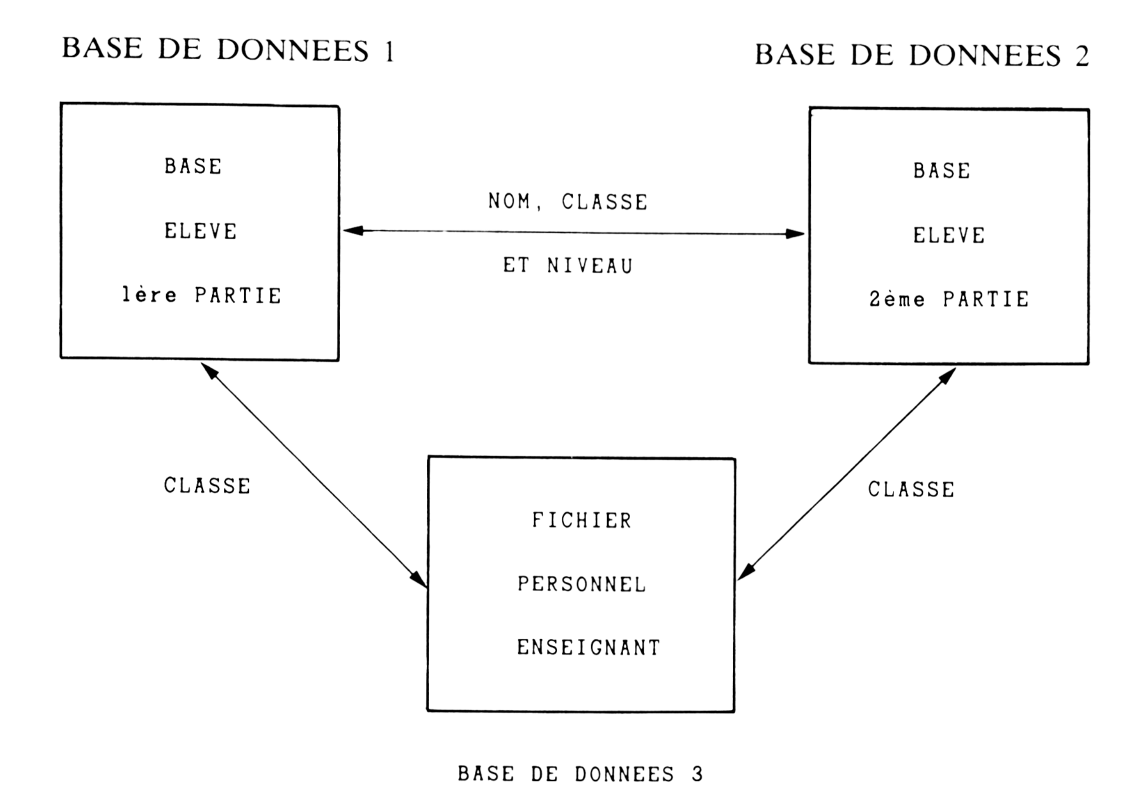


Figure 4-4

Un autre élément à considérer dans la conception est décrit par la Figure 4-5.

NOM	NOM
	PRENOM
	INITIALE DU MILIEU
ADRESSE	NUMERO
	RUE
	VILLE
	CODE POSTAL
	PAYS
TELEPHONE	CODE DEPARTEMENT
	NUMERO

Figure 4-5. Deux jeux possibles de rubriques

Cette figure éclaire un point particulier : un élément peut couvrir plusieurs informations. Dans le premier cas, nous avons trois rubriques, dans le second dix. Notre application se situera vraisemblablement entre les deux. Pour décider si l'on combine les éléments de données dans une simple rubrique ou non, il est nécessaire de savoir comment seront utilisées les données. La règle d'or est que si les éléments sont très rarement utilisés séparément, ils doivent être combinés. Le fait de regrouper les éléments tels que le prénom, le nom et l'initiale du milieu dans une simple rubrique nom, permet souvent une optimisation de l'espace. Et c'est assez facile à réaliser.

La liste de la Figure 4-5 représente exactement la même information que nous avons vue dans l'exemple de catalogue de téléphones. Il y a cependant des informations supplémentaires. Le code téléphonique et le code postal. Ces deux éléments peuvent être contenus dans les rubriques adresse et téléphone de notre exemple original.

Plus la base de données devient importante, plus l'ordinateur prend du temps pour aller rechercher l'information. Si la base de données est petite, il n'est pas nécessaire de se préoccuper de l'espace affecté à chaque rubrique. Si elle est importante, nous devons prêter attention à la taille de chaque rubrique et même à sa raison d'être.

Attention : Pour illustrer le nombre d'octets utilisés dans une rubrique, nous avons pris l'exemple du nombre de caractères dactylographiés sur une page : une page dactylographiée contient beaucoup plus d'espaces blancs pour séparer les colonnes.

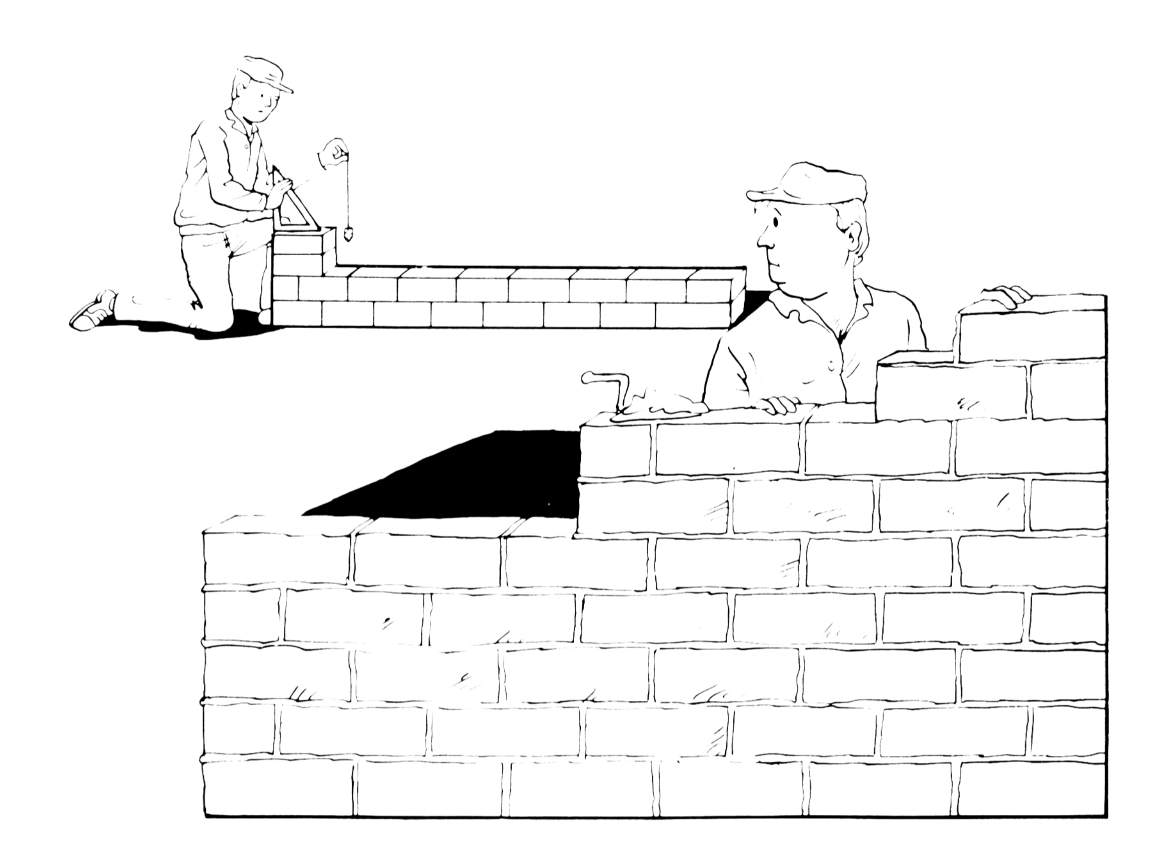
EVITEZ CELA DANS UNE BASE DE DONNEES

Les octets inutilisés ne sont pas nécessaires pour « séparer » les rubriques. Si la rubrique contient l'âge de l'élève et que l'âge maximum est 9 ans, utilisez seulement un octet pour cette rubrique. La séparation des rubriques à l'affichage, sur un terminal ou sur une imprimante, est étudiée dans les chapitres ultérieurs.

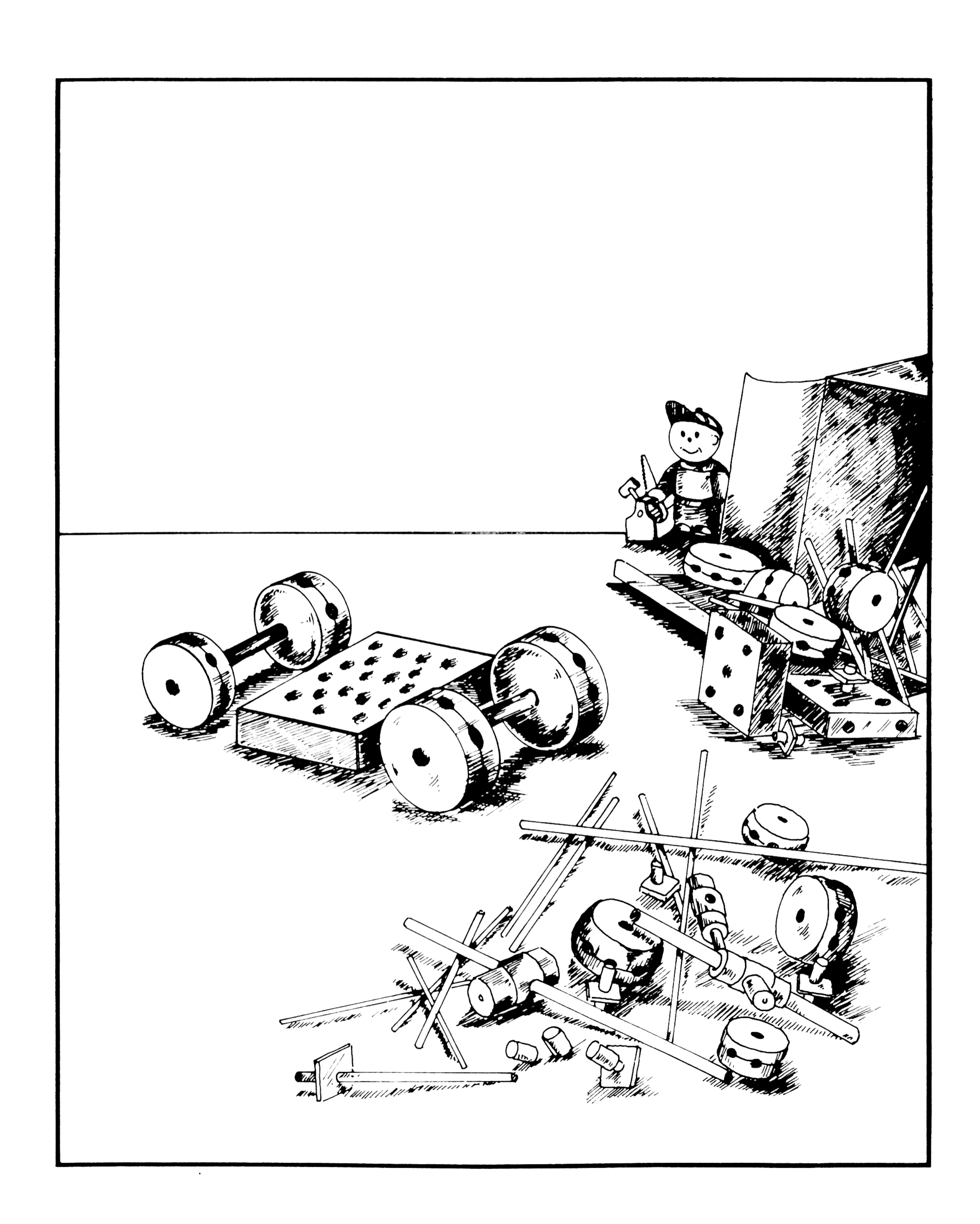
Comme nous le disions au commencement, la conception est souvent considérée comme une perte de temps. L'envie de démarrer immédiatement est la plupart du temps irrésistible. Avec l'expérience, vous apprécierez bientôt toute la valeur d'une bonne planification. La conception du projet vous oblige à penser au problème avant d'agir. Rappelez-vous que le manque de préparation de votre travail ne peut aboutir qu'à une plus grande perte de temps et vous risquez de vous retrouver dans l'obligation de repartir à zéro. Pensez à votre préparation en termes d'améliorations successives.

1) Commencez par « l'ossature » de votre plan.
2) Construisez à partir de cette structure de travail jusqu'à obtenir un système prêt à être installé sur l'ordinateur.

Le piège le plus dangereux c'est de rechercher la perfection. Cela peut se révéler coûteux et vous empêcher de réussir le plan d'une base de données. Il faut donc éviter cette attitude.



Rappelez-vous, vous n'avez pas besoin d'être familiarisé avec les systèmes de gestion de base de données sur ordinateur, les concepts, de construction et d'utilisation d'une base de données informatique sont des choses aisées. Vous connaissez certainement beaucoup d'exemples similaires qui s'exécutent avec un crayon et du papier. Détendez-vous et établissez les relations nécessaires avec les acquits de votre propre expérience. Une base de données d'ordinateur est un objectif facile à atteindre pour votre avenir.



CHAPITRE V

CONSTRUCTION DE VOTRE BASE DE DONNEES

Une fois le projet terminé, vous êtes prêt à démarrer la construction de votre base de données. Cette phase de construction commence par la création de la structure (la charpente) de la base, que nous venons d'étudier au Chapitre IV. Cette phase se terminera par la saisie de toutes les données — enregistrement par enregistrement dans la structure. Le seul problème qui peut se poser sera de ne pas trop vous ennuyer pendant la saisie de données.

Avant le développement des micro-ordinateurs et des systèmes de gestion de base de données, le processus de création d'un fichier n'était pas chose facile. C'était un processus long et coûteux qui mettait en jeu un matériel cher et le recours à des programmeurs professionnels. Bien entendu, vous aviez la possibilité de programmer vous-même, mais jusqu'à une date récente, il n'y avait aucun moyen de se soustraire à l'utilisation de matériels coûteux. Aujourd'hui, avec un système courant de gestion de base de données et un matériel micro-informatique bon marché, vous pouvez facilement faire cela vous-même. Et, à moins d'être une secrétaire très lente, cela peut se faire rapidement.

Un retour au chapitre I

Le processus de saisie de données a été illustré dans le Chapitre I par la construction de deux exemples de bases de données, B :LISTTEL et B :STOCK. Dans ces exemples, nous avons appris que la création de structure de la base de données comporte deux parties.

PREMIEREMENT, choisir un NOM DE FICHIER (titre) pour la base de données.

DEUXIEMEMENT, définir chacune des RUBRIQUES (colonnes) de la base de données.

Les règles de choix d'un NOM DE FICHIER sont déterminées par le système d'exploitation de l'ordinateur. Des micro-ordinateurs différents peuvent utiliser des systèmes d'exploitation divers. C'est pourquoi, les règles particulières de choix de noms de fichiers peuvent varier d'un ordinateur à l'autre. Comme les exemples de cet ouvrage utilisent CP/M, toutes les opérations sur les noms de fichiers seront étudiées à partir des règles de CP/M. Si vous possédez, ou vous allez acquérir un ordinateur qui n'utilise pas CP/M, il vous faudra consulter votre manuel d'instructions pour les règles de nom de fichier.

Un nom de fichier CP/M doit comporter de une à huit lettres et chiffres. Il doit commencer par une lettre. Il ne doit pas contenir d'espaces. Voilà quelques exemples de noms de fichiers corrects :

CHAPITRE
ECOLE
ANNUAIRE

Exemples de fichiers non corrects :

CHAPITRE 1	Trop long et contient un espace vierge
FACTURATION	Trop long
8TEXT	Commence par un chiffre

Pour l'ordinateur, l'objectif du nom de fichier est d'identifier avec lequel vous désirez travailler. Si l'ordinateur possède plus d'un lecteur de disque, vous devez également ajouter l'identificateur du lecteur au début du nom de fichier. Avec CP/M, les lecteurs de disque sont identifiés par une lettre suivie par deux points (par exemple, A :). Les noms de fichiers corrects ci-dessus peuvent être identifiés par :

A :CHAPITRE	Le fichier est sur le lecteur « A »
C :ECOLE	Le fichier est sur le lecteur « C »
B :ANNUAIRE	Le fichier est sur le lecteur « B »

L'identificateur de lecteur de disque n'est pas une partie indissociable du nom de fichier. Cela dépend du lecteur de disque où se trouvera insérée la disquette. Par exemple, si nous retirons la disquette contenant le fichier CHAPITRE du lecteur A et l'insérons dans le lecteur B, le fichier devra être identifié par B :CHAPITRE.

Il est possible d'avoir plus d'un fichier de base de données avec le même nom pourvu qu'ils ne se trouvent pas sur le même disque. Le système ne permet pas d'avoir deux bases de données de même nom sur le même disque.

Une base de données est un fichier de type particulier, un fichier '.DBF'. Il y a d'autres sortes de fichiers avec lesquels l'ordinateur peut travailler. Au Chapitre I, nous avons travaillé rapidement avec un autre type de fichier, appelé FORMAT (fichier « .FRM »). Nous l'avions utilisé pour préparer un document de sortie dans l'exemple de stock d'une boutique d'alcool. Un fichier de base de données et un fichier format peuvent se trouver sur le même disque et avoir le même nom de fichier. Lorsqu'un fichier de base de données est créé, dBASE II ajoute automatiquement « .DBF » au nom de fichier. Lorsqu'un document est créé, « .FRM » est automatiquement ajouté au nom de fichier. « .DBF » et « .FRM » sont des exemples de types de fichiers. Le nom de fichier est déterminé par vous-même. Avec dBASE II, le type de fichier est déterminé par le système. C'est ce qui va permettre au système d'exécuter ces tâches avec toute l'efficacité voulue.

Après avoir sélectionné un nom de fichier, vous êtes prêt à définir la structure de la base de données. L'ordinateur a besoin de connaître :

(1) Combien il y a de RUBRIQUES (colonnes).
(2) Le nom de chaque colonne.
(3) La taille de chaque colonne (par exemple le nombre de positions numériques ou caractères).
(4) Le type d'information dans chaque colonne (numérique, caractère, ou logique).

Le système de gestion de base de données vous suggèrera (question) ce dont il a besoin pour chaque élément. Un exemple de dialogue pour créer la base de données B :STOCK est présenté à l'Ecran 5-1.

```
.CREATE
ENTREZ LE NOM DE FICHIER : B:STOCK
ENTREZ LA STRUCTURE DE L'ENREGISTREMENT COMME SUIT :
 CHAMP      NOM,TYPE,DIM,DECIMALE(S)
 001        ALCOOL,C,10
 002        MARQUE,C,20
 003        CONT,C,7
 004        QUANTITE,N,3
 005        PXRV,N,6,2
 006        PXVT,N,6,2

VOULEZ-VOUS COMMENCER LA SAISIE (Y/N) ? N
```

Ecran 5-1

Nous venons de créer une base de données dont le nom de fichier est STOCK et qui est située sur le lecteur de disque B. Pour comprendre ce que nous venons de faire, prenons un classeur, écrivons « STOCK » sur la tranche, écrivons des titres de colonnes sur une feuille de papier en long, tirons des lignes sur la page pour séparer les colonnes, plaçons la feuille de papier dans le classeur, et ensuite plaçons le classeur dans une armoire.

Même processus : quelques nouvelles variables

En saisissant l'information présentée par l'Ecran 5-1, il est possible que vous tapiez une erreur. Par exemple, supposons que vous ayez mal orthographié le mot CONT dans la rubrique 3. Si vous remarquez une erreur avant d'être passé à la rubrique 4, vous pouvez la corriger. Sur la plupart des claviers, il y a une touche marquée soit RUB soit DEL. La pression sur cette touche effacera le caractère se trouvant à gauche du curseur et déplacera le curseur d'un espace vers la gauche. Une erreur de frappe peut être corrigée en effaçant toute l'écriture jusqu'à l'endroit où elle se trouve et il suffit ensuite de retaper l'écriture à partir de ce point.

Si vous remarquez une erreur après vous être déplacé vers une autre rubrique — vous ne pouvez pas revenir pour la corriger. Continuez à définir les rubriques suivantes. Pressez la touche N lorsque l'ordinateur vous demande si vous voulez commencer la saisie.

Tapez alors MODIFY STRUCTURE après le point. Un affichage semblable à l'Ecran 5-2 apparaît sur votre terminal. L'erreur peut alors être corrigée en faisant descendre le curseur vers la rubrique incorrecte, en retapant l'information, puis en tapant Ctrl W. (W pour 'write' — cela permet de sortir du mode Modification, et de l'enregistrer sur le disque).

```
                    NOM        TYPE       LONG       DEC
CHAMP 001 : ALCOOL         C         010       000       :
CHAMP 002 : MARQUE         C         020       000       :
CHAMP 003 : CONT           C         007       000       :
CHAMP 004 : QUANTITE       N         003       000       :
CHAMP 005 : PXRV           N         006       002       :
CHAMP 006 : PXVT           N         006       002       :
CHAMP 007 :                                              :
CHAMP 008 :                                              :
CHAMP 009 :                                              :
CHAMP 010 :                                              :
CHAMP 011 :                                              :
CHAMP 012 :                                              :
CHAMP 013 :                                              :
CHAMP 014 :                                              :
CHAMP 015 :                                              :
CHAMP 016 :                                              :
```

Ecran 5-2

Pour commencer à saisir les données dans notre nouvelle base de données, nous devons d'abord ouvrir le fichier. Avec dBASE II, cela se fait par l'utilisation de la commande USE. Dans notre exemple, nous aurons :

```
.USE B:STOCK
```

Le processus de saisie de données est déclenché par la commande APPEND.

```
.APPEND

ENREGISTREMENT     00050

ALCOOL    :                    :
MARQUE    :                              :
CONT      :             :
QUANTITE  :       :
PXRV      :           :
PXVT      :           :
```

Ecran 5-3

L'ordinateur vous suggère de saisir les données du premier enregistrement comme il est montré à l'Ecran 5-3. La commande APPEND ajoute alors les enregistrements à la fin de la base de données. Si l'on a déjà 49 enregistrements dans la base de données, la réponse à une commande APPEND sera identique à l'Ecran 5-3 sauf pour le numéro d'enregistrement qui devient 00050.

Après avoir saisi toutes les données d'un enregistrement, comme il est montré à l'Ecran 5-4,

```
ENREGISTREMENT     00050

ALCOOL    :  SCOTCH         :
MARQUE    :  LONG JACK                :
CONT      :  90 CL  :
QUANTITE  :  23 :
PXRV      :  39.00 :
PXVT      :  45.70 :
```

Ecran 5-4

l'ordinateur effacera complètement l'écran et vous suggèrera d'entrer les données de l'enregistrement suivant.

L'entrée des données

La valeur d'une base de données dépend pour une large part de sa qualité. Il serait peu rassurant de dire à son comptable, « Je sais, il y a quelques erreurs, mais c'est tellement rapide ! ». Il est tentant et de pratique courante, la saisie de données étant souvent une tâche fastidieuse et répétitive, d'abandonner ce travail à une personne sans qualification et peu payée. C'est une MAUVAISE idée.

Dans beaucoup de cas, il y aura une quantité énorme de données à entrer dans la nouvelle base de données. La plupart des boutiques d'alcools, par exemple, ont généralement plus de 15 articles en stock. La saisie sera longue et risque de prendre la plupart du temps disponible pour l'usage de cette base de données. La saisie est faite à une vitesse humaine ; la recherche se fait à la vitesse de l'ordinateur.

De plus, c'est la partie du processus qui est la plus sujette aux erreurs. Saisir un simple enregistrement, ou même plusieurs, peut être fait sans aucune erreur. Mais il y a peu de chances pour qu'il n'y ait pas d'erreur après l'entrée d'une centaine ou peut-être un millier d'enregistrements.

Les erreurs lors de la saisie de données

Lorsque vous ajoutez un grand nombre de nouveaux enregistrements à la base de données, lors de la mise en route par exemple, il est vraisemblable que quelque chose détournera votre attention et vous fera perdre le fil de ce que vous étiez en train de saisir. Ou, si vous êtes absorbé par le travail, vous pouvez faire une erreur pendant la saisie du dernier enregistrement. Vous ne pouvez pas par exemple revenir en arrière pour modifier ce qui est incorrect. Vous devez sortir de la commande APPEND. Vous y parviendrez en appuyant sur la touche RETURN au début de la première rubrique. Si l'enregistrement sur lequel vous travaillez, est correct, mais vous réalisez qu'une erreur a dû se glisser dans l'enregistrement précédent, continuez votre saisie des données et sortez de la commande APPEND au début de l'enregistrement suivant. Si l'enregistrement sur lequel vous travaillez n'est pas correct, vous pouvez l'annuler en appuyant sur Ctrl Q, ce qui vous permettra non seulement de sortir du mode APPEND mais également d'effacer l'enregistrement erroné que vous venez de créer.

Après être sorti du processus APPEND, vous pouvez examiner le dernier enregistrement en utilisant soit DISPLAY soit EDIT. EDIT affichera le dernier enregistrement que vous venez de saisir suivant le même principe que APPEND. EDIT vous permet également de corriger toute erreur dans l'enregistrement.

Pour pouvoir maintenant corriger les erreurs, vous avez besoin de comprendre les mouvements du curseur. Vous avez la possibilité de positionner le curseur à l'emplacement de l'erreur avant de la corriger.

Les erreurs peuvent être facilement corrigées en déplaçant le curseur en arrière vers la position de l'erreur et de taper par dessus l'information correcte. Sur beaucoup de terminaux, il y a quatre touches avec des symboles en forme de flèches. Ces touches vous permettent de déplacer le curseur pendant que vous saisissez les données dans la base. Les flèches « haut » et « bas » permettent de déplacer le curseur une rubrique en arrière ou en avant dans l'enregistrement. Les flèches « droite » et « gauche » vous permettent de déplacer le curseur d'un espace caractère vers la droite ou vers la gauche.

Certains terminaux ne disposent pas des touches de contrôle du curseur ou bien les touches ne sont pas opérationnelles avec le système de base de données. Si c'est le cas, il vous faudra utiliser une touche CONTROLE + soit E, S, D ou X pour déplacer le curseur. Vous vous rappelez de notre discussion au Chapitre I à propos de la touche CONTROLE (CTRL) : elle permet à la plupart des touches caractère du clavier d'avoir une « troisième signification ».

Avec dBASE II, cette troisième signification fournit un contrôle du curseur et d'autres fonctions pour modifier les données — comme nous allons le voir.

Le curseur se déplace à l'aide de la touche CONTROLE. Le symbole ^ signifie la touche CONTROLE. ^D veut dire : appuyer sur la touche CONTROLE et la garder enfoncée pendant que vous appuyez sur la touche D. Cette action, avec dBASE II, déplacera le curseur d'un espace vers la droite. De la même façon, ^S déplacera le curseur d'un espace vers la gauche. ^E déplace le curseur d'une rubrique en arrière jusqu'au début de l'enregistrement. ^X déplace le curseur d'une rubrique en avant. La touche CONTROLE, combinée avec les touches E, S, D et X, serviront à déplacer le curseur tout autour pour corriger les erreurs ou simplement modifier l'écriture suivant le cas.

Maintenant que vous savez déplacer le curseur, nous pouvons corriger les fautes que nous avons faites. Rappelez-vous, EDIT vous fait revenir dans les enregistrements déjà saisis et vous permet de faire des corrections. CONTROLE W « écrira » ces modifications sur le disque.

Corriger une erreur est une chose simple si vous la repérez à temps. L'ordinateur est très tolérant sur les erreurs de frappe, il vous permet de les corriger jusqu'au moment où les données sont stockées de façon permanente. L'ordinateur n'enregistre pas vos données lorsque vous passez d'un enregistrement pour en saisir un autre. Le processus de sortie fixe l'information. Vous pouvez vous déplacer à tout endroit de l'écran et modifier tout ce que vous voulez, AUSSI LONGTEMPS QUE VOUS RESTEZ DANS L'ENREGISTREMENT. Il est par conséquent suggéré de contrôler l'enregistrement qui vient d'être saisi avant de passer au suivant.

Passons à un exemple de corrections. Supposez que vous venez de taper TESSTT alors que vous vouliez TEST. Le T final peut être supprimé en plaçant le curseur sur le dernier T et en appuyant sur la barre d'espace. Le S supplémentaire peut être supprimé de deux façons : en plaçant le curseur sur le second S et en appuyant sur la touche DEL ou CLR ou en utilisant CONTROLE G (^G). DEL élimine le caractère (et l'espace qu'il occupait) à la gauche du curseur. De la même façon, CONTROLE G élimine le caractère situé sous le curseur. Dans les deux cas, nous avons comme résultat le mot TEST.

Bien entendu, nous aurions pu taper le mot TEST par dessus l'ancien mot et utiliser la barre d'espacement pour effacer les deux caractères supplémentaires à la fin. Les caractères d'un clavier conventionnel peuvent être affichés sur l'écran à l'endroit du curseur. L'ordinateur ne s'occupe pas de savoir s'il y avait des caractères ou pas. C'est le dernier caractère frappé qui « prime ».

Une autre façon de supprimer des caractères est d'en ajouter. Supposons que vous ayez tapé TET alors que vous vouliez TEST. Vous voulez insérer la lettre S entre le E et le T. Vous pouvez le faire en plaçant le curseur sur le dernier T. Pour insérer la lettre S, appuyez sur ^V, tapez S, et ensuite ^V.

^V est la commande d'« insertion ». C'est ce qu'on appelle une commande « à bascule ». Vous connaissez certainement les interrupteurs qui sont soit « allumés » soit « éteints ». C'est la même chose pour une commande à bascule qui représente un état soit 'ON' soit 'OFF'. Le même ^V allume et éteint. S'il est allumé et que vous tapiez ^V, il s'éteindra, et vice versa. Lorsqu'il est allumé, vous pouvez insérer autant de caractères que nécessaire. Il restera allumé tant que vous n'aurez pas fait d'autres ^V pour l'éteindre.

Il y a un autre élément dans la saisie de données — en dehors de la correction d'erreur — qui vous concerne. Il est possible que vous n'ayez qu'une partie de l'information nécessaire pour terminer un enregistrement, mais vous désirez tout de même saisir ce dont vous disposez.

Si vous voulez simplement entrer une partie de l'information d'un enregistrement, saisissez l'information et appuyez ensuite sur ^C. Cela vous permettra d'avancer à l'enregistrement suivant immédiatement sans être obligé de vous arrêter sur toutes les rubriques suivantes.

Dans les paragraphes précédents, vous avez remarqué plusieurs fonctions utiles par des touches de CONTROLE (^G = suppression d'un caractère ; ^V = insertion ; ^Q = je veux me débarrasser des données de cet enregistrement).

Vous pouvez retirer beaucoup d'avantages de l'utilisation des touches de CONTROLE. Les exemples fournis dans ces paragraphes doivent vous servir à acquérir une idée d'ensemble ainsi que des détails pratiques sur les commandes.

Jusqu'à maintenant, le processus de construction d'une base de données est purement mécanique. Vous créez la structure d'un fichier qui se conforme à des règles simples et vous saisissez ensuite les données. L'entrée de données se poursuit jusqu'à la saisie complète de toutes les données. A partir de là, la base de données est prête pour remplir un certain nombre de services — tels que la fourniture d'informations nécessaires aux tâches de gestion.

S'il y a peu de saisie, l'approche mécanique simple décrite ci-dessus est probablement la meilleure façon d'opérer. C'est une opération simple et directe.

S'il y a énormément d'informations à saisir, il serait bon de rechercher les moyens de s'aider de l'ordinateur dans le processus de saisie.

Assistance de l'ordinateur dans la saisie de données

Il y a un certain nombre d'aides disponibles pour la saisie de données et qui permettent à l'ordinateur de vous assister ou de réaliser à votre place l'entrée des données. Dans les pages suivantes de ce chapitre, nous envisagerons trois de ces procédés :

1. Les MESSAGES PREPARES
2. La MISE DE LA RECOPIE ON ou OFF
3. Les SYSTEMES DE MENU

Les MESSAGES PREPARES et les SYSTEMES DE MENU sont développés par des procédures simples que vous apprendrez à rédiger pour répondre à vos besoins. Une

« procédure » est un moyen facile de faire exécuter automatiquement par l'ordinateur un certain nombre de tâches spécialement pour vous. Les détails de rédaction d'une procédure seront vus dans la quatrième partie qui commence avec le Chapitre XI. L'utilisation d'une procédure qui génère des produits finis — messages préparés et systèmes de menu — est envisagée ici puisqu'elle permet d'améliorer considérablement le processus d'entrée des données. « La MISE EN RECOPIE ON ou OFF » est une aide intrinsèque, déclenchée par une simple commande.

Messages préparés

Les MESSAGES PREPARES illustrent les moyens d'assistance de l'ordinateur. Dans notre exemple de boutique d'alcools, les noms de rubriques décrivent, de façon significative, le contenu de la rubrique. C'est souvent (mais pas toujours) le cas. Si ce n'est pas le cas, il serait alors agréable de pouvoir disposer de plus d'informations. Nous pourrions fabriquer par exemple un message qui se présenterait comme suit :

```
ENTREZ LE TYPE D'ALCOOL (SCOTCH, WHISKEY, ETC) :
```

C'est beaucoup plus descriptif que le simple affichage du mot ALCOOL. En général, ce type d'aide de l'ordinateur est très apprécié. Elle est particulièrement utile si vous voulez être assisté dans la saisie d'un grand nombre de données. C'est particulièrement vrai si les noms de fichiers ne sont pas réellement significatifs de leur contenu interne.

Comme nous l'avons mentionné précédemment, les messages préparés tels que celui-ci peuvent être produits relativement aisément à partir des PROCEDURES.

Les procédures, comme les bases de données, ont des noms de fichiers. Avec dBASE II, les procédures sont appelées des fichiers de COMMANDE. Les règles qui s'appliquent aux autres fichiers s'appliquent également aux fichiers de COMMANDE. Vous devez également identifier le lecteur de disque sur lequel est rangée la procédure.

Pour vous montrer comment fonctionnent les MESSAGES PREPARES, nous admettrons que l'on vient d'écrire une procédure décrivant des messages. Nous appellerons notre exemple de procédure B :SAISIE. Pour faire exécuter cette procédure par l'ordinateur avec dBASE II, vous tapez :

```
.DO B:SAISIE
```

DO B :SAISIE fonctionne exactement comme APPEND. APPEND vous fournissait le format d'un enregistrement « standard » sans aucune donnée ; B :SAISIE vous présente le format d'un enregistrement « préparé » par vous et vide. DO B :SAISIE produira l'Ecran 5-5.

```
ENREGISTREMENT # 00001

ENTREZ LE TYPE D'ALCOOL (VODKA,GIN,ETC.)     :               :
ENTREZ LE NOM DE LA MARQUE                   :                   :
ENTREZ LA CONTENANCE EN CL (90,100,120,ETC.) :          :
QUANTITE DANS CETTE CONTENANCE ET MARQUE     :      :
PRIX DE REVIENT       :         :
PRIX DE VENTE       :         :
```

Ecran 5-5

Note : La rédaction de procédures avec un système de gestion de base de données n'est pas une chose difficile. En réalité, cela peut être agréable et en valoir la peine.

La mise en recopie ON ou OFF

Nous allons ouvrir une petite parenthèse. Dans beaucoup de bases de données, il y a souvent une grande majorité d'informations identiques à saisir. Dans une école, il y a beaucoup plus d'élèves que de classes et de professeurs. Dans notre exemple de boutique d'alcools, il y a plusieurs noms de marques pour chaque type d'alcool. Dans beaucoup de cas, les données peuvent être groupées de manière à réduire la quantité d'informations qui doivent être tapées. Dans la boutique d'alcools, le stock sur les rayons est groupé par type d'alcool — Vodka, Bourbon, etc. — pour la commodité du client. Les contenances variées pour une marque donnée sont généralement regroupées par marque.

Lorsque la redondance est groupée, comme dans ces deux exemples, l'ordinateur peut réduire la quantité de frappe requise en recopiant les données précédentes d'un enregistrement sur l'autre. Avec dBASE II, cette possibilité est « déclenchée » par la commande SET CARRY ON. On l'éteint par SET CARRY OFF.

Voyons comment l'information saisie progresse dans notre exemple si CARRY a été mis ON. Avec APPEND, nous obtenons d'abord un premier affichage écran à partir duquel nous entrons les données de l'enregistrement 00001.

```
ENREGISTREMENT # 00001

ALCOOL       : SCOTCH       :
MARQUE       : LONG JACK          :
CONT         : 90  :
QUANTITE     : 23 :
PXRV         : 39.00 :
PXVT         : 45.70 :
```

Ecran 5-4

Normalement, les emplacements de contenu des rubriques apparaissent vierges, avec le titre ENREGISTREMENT 00002. Avec CARRY ON, l'affichage est exactement le même que l'écran 5-4 avec le numéro d'enregistrement avancé à 00002. Tout ce que nous avons à faire est de modifier les rubriques qui diffèrent de l'enregistrement 1. Lorsque nous avançons, l'affichage de l'enregistrement 3 sera exactement comme celui du numéro 2. Cette aide peut nous servir à deux choses : réduire considérablement la quantité de frappe ainsi que le nombre d'erreurs dues à la frappe.

Un système de menu

Maintenant, revenons à une autre technique qui permet d'« apprendre » à l'ordinateur à faire. Avec cette technique, on rédige une procédure qui fournit à l'opératrice la possibilité de choisir une fonction à partir de « choix multiples ». Cette technique s'appelle un SYSTEME DE MENU et aboutira à minimiser les erreurs de frappe.

Malheureusement, elle augmente les possibilités d'erreur absolue. On s'en sert surtout lorsque les données n'ont rien de commun d'un enregistrement à l'autre et, par voie de conséquence, la recopie d'un enregistrement à l'autre ne sert pas à grand chose.

Supposons que nous ayons écrit une procédure établissant le stock d'une boutique d'alcools avec la technique du MENU. Dans notre exemple hypothétique, le terminal donnerait l'affichage de l'Ecran 5-6.

```
                CHOISISSEZ PARMI LES ELEMENTS SUIVANTS

     A - 1/2 L                          1 - SCOTCH
     B - 60 CL                          2 - BOURBON
     C - 70 CL                          3 - GIN
     D - 90 CL                          4 - WHISKEY
     E - LITRE                          5 - VODKA
     F - 120 CL                         6 - WHISKEY IRLANDAIS
     G - 1 L 1/2                        7 - RHUM
     H - 2 L                            8 - BRANDY
     I - CAISSE                         9 - AUTRES

ENTREZ LA CONTENANCE CHOISIE DE LA COLONNE GAUCHE :   :

ENTREZ LE TYPE DE SELECTION D'ALCOOL DE LA COLONNE DE DROITE :   :

ENTREZ LE NOM DE LA MARQUE :                :

ENTREZ LA QUANTITE :  :

ENTREZ LE PRIX DE REVIENT :  :

ENTREZ LE PRIX DE VENTE :  :
```

Ecran 5-6

Cet exemple exploite bien l'aspect utilitaire des messages préparés et la sélection par un menu dont nous avions parlé auparavant. L'information « 2 L » de gin correspondra à H3 dans la réponse.

Dans cet exemple, on utilise un seul écran pour saisir et proposer le menu. S'il y a beaucoup plus de données, nous aurons besoin d'un écran supplémentaire. Si nous avons à choisir par exemple parmi 30 ou 40 types d'alcools et contenances, nous pourrions utiliser un écran pour chacune des colonnes de sélection. Ce qui nous amènerait à trois écrans successifs dans cet exemple de menu.

Les systèmes qui utilisent les menus comme les systèmes de message, peuvent être exploités utilement lorsque vous avez beaucoup de données à saisir sans montrer la nature de l'information ou quoi que ce soit sur l'organisation du système de base de données. Il y a des cas où tout ce qui doit être saisi peut être choisi dans un menu.

Remplacement conditionnel

En plus de ces utilitaires directs pour la saisie, on peut recourir à une méthode plus astucieuse pour entrer un certain type d'information « conditionnelle ».

La base de données de l'école élémentaire est un bon exemple pour la méthode de REMPLACEMENT CONDITIONNEL.

Cette base de données contient le NOM, la CLASSE, le NIVEAU et le PROFESSEUR de chaque élève. Lorsque l'on construit la base, seule la saisie du NOM, de la CLASSE et du NIVEAU est nécessaire. Lorsque toutes les informations de chaque élève auront été saisies, l'information PROFESSEUR peut être ajoutée avec la commande dBASE II REPLACE.

```
.REPLACE PROF WITH 'M. MARTIN' FOR CLASSE='101'
.REPLACE PROF WITH 'MLLE LISE' FOR CLASSE='201'
ETC
```

Lorsqu'on construit une base de données, il est conseillé de faire la saisie en une seule opération. Ce processus de saisie s'appelle le CHARGEMENT de la base de données. Le processus de « construction » est primordial dans l'utilisation ultérieure de la base de données. Si l'information est incomplète ou si elle est saisie avec des

erreurs, le résultat final sera incomplet et faux. Vous serez alors déçu. Après avoir entré correctement les données, vous pouvez commencer à les utiliser de façon profitable — ce qui est finalement l'objectif.

Les options de construction de la base de données originale vont du simple et direct APPEND, à des procédures plus élaborées fournissant des messages d'explications plus clairs. Les autres moyens servant à minimiser votre effort sont :

- recopier l'information d'un enregistrement sur l'autre,
- la sélection par un MENU, et
- le REMPLACEMENT CONDITIONNEL.

Ces utilitaires réduisent le volume de frappe. Plus votre base de données sera importante et complexe plus l'idée de procédure sera intéressante pour vous.

CHAPITRE VI

MODIFIER ET REORGANISER VOTRE BASE DE DONNEES

La base de données une fois construite sera inévitablement modifiée. Tous les types de modifications aboutissent à la modification de l'enregistrement. En plus des changements dans les rubriques telles que QUANTITE et PRIX, certains enregistrements doivent être ajoutés et d'autres supprimés. Le développement de nouvelles réglementations peut avoir pour conséquence l'ajout de nouvelles rubriques à la base de données. Cette activité courante de modification de la base s'appelle LA MISE A JOUR.

La mise à jour est une activité qui peut prendre beaucoup de temps. C'est particulièrement dû au fait qu'il s'agit d'une opération manuelle. La routine de sortie d'états imprimés ou autres est effectuée automatiquement à la vitesse de l'ordinateur et nécessite relativement peu de temps.

La fréquence avec laquelle vous mettrez à jour dépendra en grande partie de vos besoins. Vous serez obligé de faire quotidiennement un certain nombre de tâches de mise à jour. D'autres pourront se dérouler toutes les semaines, tous les mois, etc. Les autres tâches de mise à jour se feront seulement si besoin est.

Notre petite boutique d'alcools, par exemple, peut mettre à jour la base de données de stock à chaque nouvelle livraison. Le relevé d'heures des employés peut être mis à jour, suivant le cas, quotidiennement ou toutes les semaines. L'importance des tâches de mise à jour dépendra bien entendu de l'application en question.

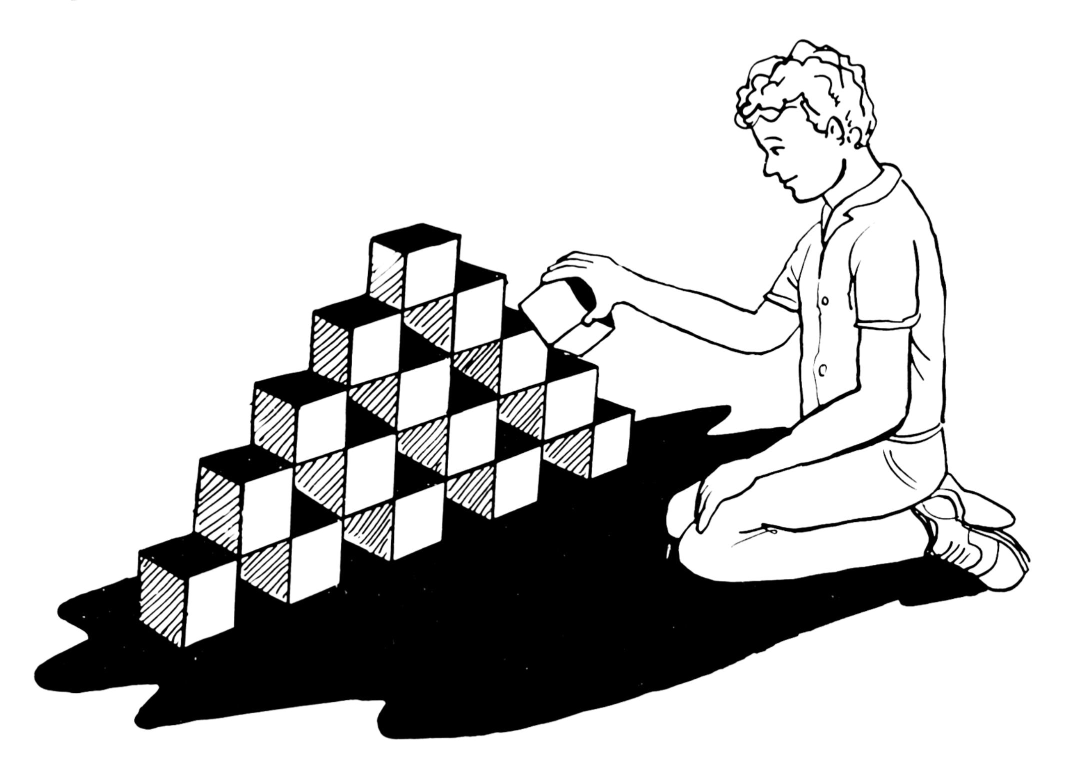
On distingue au moins trois catégories de modifications :

(1) Modifier la structure de la base de données
(2) Ajouter et supprimer des enregistrements
(3) Modifier le contenu des enregistrements.

Modifier la structure d'une base de données

Généralement, la structure d'une base de données reste assez longtemps inchangée. La modification de structure est généralement une réponse aux modifications dans l'environnement de travail — telles que de nouvelles réglementations. Sous peine de perte de données, les modifications de structure de la base doivent être faites avec le plus grand soin.

Lorsque l'on change la structure d'une base (MODIFY), son contenu est détruit. On doit faire une copie des données de la base avant de modifier sa structure. Après cette copie, la structure peut être modifiée en toute sécurité.



Un exemple de ce processus avec le système dBASE II est représenté par l'Ecran 6-1.

```
.USE B:STOCK
.COPY ALL TO B:NOUVEAU
00015 ENREGISTREMENT(S) TRANSFERE(S)
.MODIFY STRUCTURE
LA COMMANDE MODIFY DETRUIRA TOUTES VOS DONNEES (Y/N) ? Y
```

Ecran 6-1

Les modifications de structure possibles comprennent :

- L'ajout de rubriques
- La suppression de rubriques
- La modification du nom d'une rubrique (opération spéciale)
- La modification de la taille d'une rubrique et
- La modification du type d'une rubrique

L'ordinateur affichera la structure de la base de données sur le terminal comme à l'Ecran 6-2.

	NOM	TYPE	LONG	DEC	
CHAMP 01 :	ALCOOL	C	010	000	:
CHAMP 02 :	MARQUE	C	020	000	:
CHAMP 03 :	CONT	C	007	000	:
CHAMP 04 :	QUANTITE	N	003	000	:
CHAMP 05 :	PXRV	N	006	002	:
CHAMP 06 :	PXVT	N	006	002	:
CHAMP 07 :					:
CHAMP 08 :					:
CHAMP 09 :					:
CHAMP 10 :					:
CHAMP 11 :					:
CHAMP 12 :					:
CHAMP 13 :					:
CHAMP 14 :					:
CHAMP 15 :					:
CHAMP 16 :					:

Ecran 6-2

Dans ce mode, vous êtes capable de déplacer le curseur d'un bout à l'autre de l'écran pour effectuer les modifications souhaitées. Le curseur peut être déplacé en utilisant les touches de déplacement de curseur. Si votre clavier ne dispose pas de ces touches, le curseur peut être déplacé en utilisant la touche CONTROLE comme il est montré à la Figure 6-1. Une rubrique peut être supprimée en positionnant le curseur sur le nom de rubrique et en appuyant soit sur CONTROLE Y soit CONTROLE T. Une rubrique peut être ajoutée en positionnant le curseur à l'endroit voulu de la nouvelle rubrique et en appuyant sur CONTROLE N. Ce qui aura pour effet de déplacer toutes les rubriques suivantes d'une position vers le bas et d'afficher un espace vierge pour l'emplacement d'une nouvelle rubrique. Tapez alors le NOM, le TYPE et la TAILLE DE LA RUBRIQUE. Les rubriques existantes peuvent être

modifiées en positionnant le curseur sur la rubrique à modifier et en tapant la nouvelle information sur l'ancienne. En tapant CONTROLE W, vous indiquez à l'ordinateur que vous venez de terminer les modifications de structure. CONTROLE Q annule cette modification.

TOUCHE DE CONTROLE	EFFET
Ctrl - S ou ←	Déplace le curseur d'un caractère vers la gauche
Ctrl - D ou →	Déplace le curseur d'un caractère vers la droite
Ctrl - E ou ↑	Déplace le curseur à la rubrique précédente
Ctrl - X ou ↓	Déplace le curseur à la rubrique suivante
Ctrl - T	Supprime l'enregistrement
Ctrl - Y	Supprime ce qui vient d'être tapé
Ctrl - N	Insère un espace vierge pour une nouvelle rubrique à l'emplacement du curseur
Ctrl - W	Sauvegarde la structure du nouveau fichier
Ctrl - Q	Annule l'opération de modification

Figure 6-1. Les fonctions de modification de structure avec la touche de controle (MODIFY STRUCTURE)

A ce stade, nous avons une copie de l'ancienne base de données et une nouvelle base qui n'a pas d'enregistrements. Les données de l'ancienne base sont CHARGEES (RECHARGEES) dans la nouvelle base de données par la commande dBASE II APPEND.

```
.APPEND FROM B:NOUVEAU
00015 ENREGISTREMENT(S) TRANSFERE(S)
```

Les contenus de toutes les rubriques qui sont communes à la fois à la nouvelle et à l'ancienne base de données seront ajoutés à la nouvelle. Les nouvelles rubriques se présenteront comme vides. Une rubrique dont on a rétréci le champ tronquera les données lorsqu'elles seront ajoutées à la nouvelle base. Les rubriques caractères perdront leurs caractères les plus à droite tandis que les rubriques numériques

perdront leurs chiffres les plus à gauche. Le problème se pose lorsqu'un nom de rubrique a été modifié. Dans ce cas, l'ordinateur admet implicitement que cette rubrique est une nouvelle rubrique et que l'ancienne est supprimée. Par conséquent, cette rubrique sera vide dans le nouvel enregistrement.

La modification des noms de rubriques nécessite une opération spéciale. Pour y parvenir, il faut décharger et recharger la base de données. Lorsque l'on utilise cette méthode, les noms de rubriques ne doivent pas être ajoutés ou supprimés. Toutes les tailles de rubrique doivent également rester intactes. Avec dBASE II, cette opération est accomplie avec une option de la commande COPY. (Dans certains systèmes, c'est ce que l'on appelle décharger, c'est une autre façon de copier. La base de données déchargée ne peut pas être utilisée directement par le système de gestion de la base. On utilise cette copie spéciale pour recharger le système de la base de données une fois que les modifications de structure ont été faites).

```
.COPY ALL TO B:NOUVEAU SDF
```

Cela a pour effet de stocker le contenu de la base de données dans NOUVEAU d'une manière un peu spéciale. Il n'y a plus de rubriques — bien que les données soient un tableau de lignes et de colonnes. Le système de gestion de base de données ne peut pas les utiliser directement. On peut cependant les reprendre par un système de traitement de texte. SDF veut dire Système de Format de Données. C'est un exemple de base de données DECHARGEE.

Les données déchargées qui ont été stockées dans le fichier B :NOUVEAU peuvent être chargées dans la base de données STOCK par

```
.APPEND FROM B:NOUVEAU SDF
```

La structure de STOCK est la même qu'auparavant excepté la modification d'une rubrique — la base de données est maintenant remise dans son état original — on peut raisonnablement espérer que la structure de la base de données ne soit pas modifiée constamment. Dans ce cas, il est de bon ton de faire une copie supplémentaire — simplement au cas où les choses tourneraient mal. C'est le seul cas où il y a un léger risque pour les données.

Ajouter et supprimer des enregistrements

Une opération bien plus courante est l'ajout ou la suppression d'enregistrements de la base. De nouveaux employés sont recrutés, d'autres s'en vont ou partent à la retraite. Ces opérations se font avec une grande facilité.

Le processus d'ajout d'un enregistrement à la base de données a été décrit aux Chapitres II et V (Commande dBASE II APPEND). Lorsque l'on ajoute un enregistrement, il est placé à la fin de la base de données. Avec dBASE II, la fin d'une base de données est appelée le fond (Bottom). Il y a des circonstances particulières où il est souhaitable d'insérer un enregistrement au milieu de la base de données. Avec dBASE II, on prend la commande INSERT. Cette commande est similaire à la commande APPEND excepté qu'elle ajoute le nouvel enregistrement à l'endroit désiré dans la base. Pour utiliser cette commande, vous devez vous positionner vous-même dans la base de données. On y parvient au moyen de la commande GOTO.

```
.GOTO RECORD 127
```

INSERT insérera un enregistrement vide à l'emplacement de l'enregistrement 128. Tous les enregistrements dont les numéros sont supérieurs à 127 seront renumérotés et déplacés un enregistrement plus loin. L'ancien enregistrement 128 devient l'enregistrement 129, l'ancien enregistrement 129 devient l'enregistrement 130, et ainsi de suite. Une fois cette remise en ordre terminée, l'écran s'effacera et l'enregistrement vierge 128 sera présenté pour la saisie exactement comme avec la commande APPEND.

Les enregistrements sont supprimés par un processus en deux étapes. Premièrement, un enregistrement est marqué pour effacement. Les commandes suivantes marquent un enregistrement pour effacement :

```
.DELETE RECORD 216

.DELETE FOR NOM='MARTIN,GERARD'

.DELETE FOR ALCOOL='WHISKEY' .AND. MARQUE='LONG JACK'
```

Chaque fois qu'il y a une commande d'effacement, l'ordinateur répond en indiquant le nombre d'enregistrements qui viennent d'être effacés. Ceci permet de vous corriger et éventuellement de rappeler les enregistrements qui auraient été effacés par erreur. La commande dBASE II de rappel des enregistrements est RECALL. Voici quelques exemples de rappel d'enregistrement :

```
.RECALL ALL
.RECALL RECORD 216
.RECALL FOR NOM='MARTIN,GERARD'
```

La suppression réelle des enregistrements est faite par une deuxième commande, nommée PACK. PACK supprime de façon permanente les enregistrements qui ont été marqués au moyen de la commande DELETE.

Si la base de données est utilisée en conjonction avec des tables d'index telles que les fichiers INDEX de dBASE II, lorsque les enregistrements sont ajoutés ou supprimés, la base de données doit être ré-indexée. Certains autres systèmes de gestion de base de données peuvent permettre d'ajouter ou de supprimer des enregistrements avec la présence d'un fichier d'index. Lorsque c'est le cas, le fichier d'index est automatiquement mis à jour après chaque ajout ou suppression d'enregistrement. C'est le cas avec les systèmes de base de données hiérarchiques ou en réseau (CODASYL).

Lorsqu'un enregistrement est ajouté et qu'il y a rattachement d'un fichier d'index, l'enregistrement apparaît inséré dans la base de données. Si, par exemple, la base de données est indexée sur le NOM, les enregistrements apparaîtront en ordre alphabétique. Un nouvel enregistrement apparaîtra placé à sa position alphabétique propre. Il est en réalité ajouté physiquement au fond de la base de données exactement comme s'il n'y avait pas de fichier d'index.

Modification du contenu d'un enregistrement

Il y a plusieurs façons de changer le contenu des rubriques de données. Lorsque vous effectuez des modifications particulières sur des enregistrements individuels (telles que la saisie du nombre d'heures d'un employé qui travaille le lundi), l'opération peut se dérouler au coup par coup avec la commande dBASE II EDIT. Celle-ci présente le même affichage que la commande APPEND (mode plein écran).

EDIT est la seule commande qui nécessite la connaissance du numéro d'enregistrement. Supposons que nous voulions modifier l'enregistrement du stock de notre boutique d'alcools et changer la contenance de 90 cl du scotch Long Jack. Si nous savons que lle numéro d'enregistrement de cet article est 1, nous pouvons commencer le processus de modification par :

```
.EDIT 1
```

Le terminal se présentera comme à l'Ecran 6-3. L'enregistrement peut être modifié en déplaçant le curseur sur la rubrique désirée et en tapant la nouvelle information. Le curseur peut être déplacé avec les touches de déplacement curseur. S'il n'y a pas de touches de contrôle du curseur, celui-ci doit être déplacé en utilisant la touche CONTROLE, comme indiqué à la Figure 6-2.

Pour illustrer les possibilités de EDIT, nous allons modifier la quantité de 23 par 33, le PXRV de 39.00 et le PXVT de 45.70. Nous opérerons avec la séquence suivante. Appuyez sur la touche CONTROLE. Pendant que vous la tenez enfoncée, vous appuyez trois fois sur la touche X. Cela a pour effet de déplacer le curseur de la rubrique ALCOOL à la rubrique QUANTITE. Tapez alors 33 et RETURN. Ce qui remplace la valeur 23 par 33 et déplace le curseur à la rubrique PXRV. Tapez alors 41.00 et RETURN. Ceci remplace la valeur 39.00 par 41.00 et déplace le curseur à la rubrique PXVT. Tapez ensuite 49.00 et RETURN. Ce qui remplace la valeur 45.70 par 49.00 et avance le curseur à l'enregistrement suivant — ici, l'enregistrement 2. Appuyez sur CONTROLE R pour revenir à l'enregistrement 1. Cet enregistrement est présenté par l'Ecran 6-4.

```
ENREGISTREMENT # 00001

ALCOOL     : SCOTCH      :
MARQUE     : LONG JACK            :
CONT       : 90 CL :
QUANTITE   : 23 :
PXRV       : 41.00 :
PXVT       : 49.00 :
```

Ecran 6-3

Les enregistrements adjacents peuvent être appelés par ^R (pour l'enregistrement précédent) et ^C (pour l'enregistrement suivant). L'opération peut être annulée en utilisant ^Q. Annuler les modifications peut se révéler pratique lorsque vous vous apercevez que vous êtes en train de modifier le mauvais enregistrement. Les modifications peuvent être prises en compte soit par ^W, ^R ou ^C. La commande EDIT vous permet de vous déplacer d'un enregistrement à l'autre par l'utilisation répétée de ^C.

TOUCHE DE CONTROLE	EFFET
Ctrl - S ou ←	Déplace le curseur d'un caractère vers la gauche
Ctrl - D ou →	Déplace le curseur d'un caractère vers la droite
Ctrl - E ou ↑	Déplace le curseur sur la rubrique précédente
Ctrl - X ou ↓	Déplace le curseur sur la rubrique suivante
Ctrl - Y	Supprime le contenu de l'enregistrement
Ctrl - U	Interrupteur - supprime/ne supprime pas l'enregistrement
Ctrl - V	Interrupteur - allume/éteint le mode d'insertion caractère
Ctrl - W	Sauvegarde les informations du nouveau fichier
Ctrl - Q	Annule l'opération de modification
Ctrl - R	Affiche l'enregistrement précédent
Ctrl - C	Se positionne sur l'enregistrement suivant

Figure 6-2. Les fonctions des touches de contrôle pendant la modification avec EDIT

```
ENREGISTREMENT # 00001

ALCOOL      : SCOTCH       :
MARQUE      : LONG JACK           :
CONT        : 90 CL :
QUANTITE    : 33 :
PXRV        : 43.00 :
PXVT        : 52.00 :
```

Ecran 6-4

Il peut arriver que nous ne connaissions pas le numéro d'enregistrement, on peut l'obtenir de plusieurs manières :

```
.DISPLAY FOR ALCOOL='SCOTCH' .AND. MARQUE='LONG JACK'

.LOCATE FOR ALCOOL='SCOTCH' .AND. MARQUE='LONG JACK' .AND.CONT=90 CL'
```

Chacune de ces commandes fournit le numéro d'enregistrement souhaité. Le deuxième exemple de commande « positionne » également la base de données sur l'enregistrement souhaité. Dans ce cas, EDIT modifiera l'enregistrement en cours.

L'autre méthode permettant de voir et de modifier le contenu de la base de données est illustrée par la commande dBASE II BROWSE. Un exemple graphique de BROWSE peut être représenté en découpant une fenêtre carrée dans une feuille de papier. Maintenant placez le papier avec son ouverture sur une autre feuille contenant des inscriptions. Vous obtenez une sorte de fenêtre s'ouvrant sur la seconde feuille. En déplaçant la fenêtre au fur et à mesure, vous finissez par voir le contenu entier de la seconde page. BROWSE est une fenêtre sur la base de données qui vous permet à la fois de voir une partie des données et de faire des modifications lorsque vous le souhaitez. La différence entre les fonctions BROWSE et EDIT, c'est que BROWSE vous montre une partie de plusieurs enregistrements sur l'écran à chaque fois tandis que EDIT vous montre la totalité d'un seul enregistrement.

Les commandes EDIT et BROWSE sont nécessairement limitées à des opérations manuelles. De plus, on ne peut modifier qu'un enregistrement à la fois.

Le système de base de données doit également fournir la possibilité de modifier plusieurs enregistrements en une seule opération lorsqu'une CONDITION a changé. Prenons par exemple le départ d'un professeur et le remplacement par un nouvel enseignant. L'utilisation de EDIT ou de BROWSE nécessite la frappe du nom du nouveau professeur pour chaque enregistrement qui doit être modifié. Une approche plus simple consiste à utiliser la commande dBASE II REPLACE. Si le nouveau professeur est Melle Lise, l'ancien professeur M. Martin et la classe numéro 21, la commande apparaîtra comme ci-dessous. Lorsque l'on utilise REPLACE avec une condition, chaque enregistrement qui satisfait la condition est modifié.

```
.REPLACE PROF WITH 'MELLE LISE' FOR CLASSE='21'
.REPLACE PROF WITH 'M. MARTIN' FOR PROF='M. DURANDAL'
```

Les applications de base de données qui nécessitent des modifications fréquentes bénéficieront de l'emploi des procédures illustrées au chapitre précédent : les messages descriptifs et les menus. L'utilisation de procédures vous permet de faire moins d'effort de mémoire. La base de données peut être manipulée par des personnes moins spécialisées. Les deux techniques peuvent être combinées, comme au dernier chapitre, pour fournir une aide souple et efficace dans la modification de la base.

L'Ecran 6-5 illustre l'opération de modification d'une base de données d'une école primaire à l'aide d'une PROCEDURE. Celle-ci utilise une combinaison de messages descriptifs et d'aide par un menu pour la personne qui saisit les données. Cet exemple ajoute des enregistrements, en supprime et fournit la possibilité de les modifier. Les enregistrements contiennent le NOM, la CLASSE, le NIVEAU, le PROFESSEUR et des rubriques contenant d'autres informations sur chaque élève.

```
                CHOISISSEZ DANS CE QUI SUIT :

    A - AJOUTER UN NOUVEL ELEVE

    C - MODIFIER L'ENREGISTREMENT D'UN ELEVE

    S - MODIFIER L'ENREGISTREMENT SUIVANT D'UN ELEVE

    D - SUPPRIMER L'ENREGISTREMENT D'UN ELEVE

    Q - FIN DE SESSION

S'IL VOUS PLAIT, ENTREZ VOTRE CHOIX :   :
```

Ecran 6-5

Si c'est C qui est choisi, l'ordinateur présentera un nouvel affichage. Celui-ci est montré par l'Ecran 6-6.

```
     SVP ENTREZ LES INFORMATIONS SUIVANTES

ENTREZ LE NIVEAU DE L'ELEVE  :  :

ENTREZ LE NUMERO DE LA CLASSE DE L'ELEVE  :      :

ENTREZ LE NOM DE L'ELEVE      :                          :

ENTREZ LE PRENOM DE L'ELEVE  :                                :
```

Ecran 6-6

Lorsque l'information a été fournie à l'ordinateur, celui-ci affiche un court message pendant qu'il fait la recherche de l'élève. L'ordinateur affiche alors l'enregistrement de l'élève comme sur l'Ecran 6-7.

```
                    ENREGISTREMENT DE L'ELEVE

NOM DE L'ELEVE (NOM PRENOM) : MERAUD, GEORGES :

CLASSE : 201 :   NIVEAU : 5 :   NOM DU PROFESSEUR : MARTINOT :

DATE DE NAISSANCE (JOUR/MOIS/ANNEE) : 11/04/71 :

... etc pour le reste de l'enregistrement de l'élève.
```

Ecran 6-7

On modifie les données en déplaçant le curseur sur la rubrique que l'on veut modifier et en tapant la nouvelle information. Lorsque les changements sont terminés, l'opérateur revient au Menu principal. Une nouvelle opération de modification est choisie et le processus recommence. L'option A du menu se place directement sur l'enregistrement de l'élève. Celui-ci bien entendu apparaît à l'affichage complètement vide. L'option S du menu fournit le moyen de se déplacer d'enregistrement en enregistrement, au fur et à mesure de la saisie des résultats scolaires, sans qu'il soit nécessaire d'entrer des informations inutiles. L'option D du menu utilise le deuxième affichage (la demande d'information sur l'élève). Si l'on choisit l'option d'effacement, la procédure fournit toujours une deuxième chance à l'utilisateur. Après avoir entré l'information, l'ordinateur peut fournir à ce dernier un affichage semblable à celui présenté par l'Ecran 6-8.

```
                 ENREGISTREMENT EFFACE

       VOUS AVEZ EFFACE L'ENREGISTREMENT SUIVANT

                    MERAUD, GEORGES

                NIVEAU 5     CLASSE 201

ETES-VOUS SUR DE VOULOIR SUPPRIMER CET ELEVE (O/N) :     :
```

Ecran 6-8

Cet exemple est représentatif de l'utilisation de menus et de messages descriptifs pour aider l'utilisateur à entrer de nouvelles informations dans l'ordinateur. Cette opération ressemble à l'utilisation de formulaires dans la vie courante. Dans l'administration par exemple, il existe une procédure qui décrit quel imprimé ou quel formulaire choisir et dans quel cas. Pour ce qui nous préoccupe, la procédure indique à l'ordinateur le format à utiliser. Si l'utilisateur n'est pas familier avec les ordinateurs, la procédure pourrait contenir une option « assistance de l'opérateur » ou AIDE dans son menu.

L'un des atouts supplémentaires de la procédure est de formaliser le processus de modification de la base de données. On sollicite moins d'effort de la part de l'utilisateur. Si le processus de saisie des données peut être fait de manière intéressante, on réduit le risque d'erreur. Pour la plupart des applications de base de données, il est impératif que la base ne contienne pas d'erreur. Le recours à des menus minimise l'effort et les risques d'erreurs d'orthographe (rappelez-vous que si, pour vous, scocth signifie scotch — ce n'est pas évident pour l'ordinateur). Il existe quand même un risque pour le mot GIN à la place de SCOTCH. La procédure peut fournir une sécurité contre ce genre d'erreur, cependant cela nécessitera un effort supplémentaire de la part de la personne qui écrira la procédure.

Le dernier thème abordé dans ce chapitre concerne l'entretien de la base de données. Dans ce cas, « l'entretien » peut se comprendre comme « assurer la sécurité » de la base de données. Si vous disposez d'une « base de données » qui se compose d'un tas d'enregistrements sur papier, vous serez sujet au même problème de sauvegarde des enregistrements papier. On peut répandre du café sur une partie des informations, une simple feuille de papier peut être égarée ou jetée par inadvertance.

Seuls des incidents majeurs (un feu, une inondation, une tornade, etc.) seront catastrophiques pour la « base de données ». Ce n'est pas vrai avec une base de données d'ordinateur. Toute la base peut être contenue sur une fine feuille de mylar magnétisée. Un simple doigt posé par inadvertance sur le film de mylar peut détruire l'information qui y est stockée. Une cigarette la détruira entièrement. Un morceau de film est plus vulnérable qu'une caisse remplie de papier. Toutes les choses qui servent à travailler avec cette base de données de manière commode peuvent être susceptibles de la détruire.

Bien des tâtonnements peuvent être supprimés en se conformant strictement à quelques procédures simples. Une copie ou deux de sauvegarde de la base doivent être mises à jour et protégées. C'est une des raisons pour lesquelles il est bon de disposer de plus d'un lecteur de disque dans votre configuration d'ordinateur. Si vous avez deux lecteurs, vous pouvez faire la copie facilement en plaçant le disque vierge dans l'un des lecteurs supplémentaires en utilisant la commande copy. Si votre base de données est située sur le disque B et le disque est dans le lecteur C, la copie s'effectuera par :

```
.COPY ALL TO C:BACKUP
```

Si vous avez seulement deux lecteurs, votre base est située dans le lecteur de disque B et s'il y a suffisamment de place disponible dans le lecteur A pour contenir la base de données, la copie peut être faite par :

```
.COPY ALL TO A:BACKUP
```

Si vous avez seulement deux lecteurs de disque, votre base est située dans le lecteur de disque B et s'il n'y a pas suffisamment de place disponible dans le lecteur A pour contenir la base de données, utilisez la commande dBASE II QUIT. QUIT vous permettra de revenir au système d'exploitation de votre ordinateur.

```
.QUIT

A>
```

Si le lecteur A supporte une copie du CP.M, votre système d'exploitation, la procédure suivante vous fournira une copie de sauvegarde de votre base de données. Retirez le disque contenant votre base du lecteur de disque B. Insérez une disquette vierge dans le lecteur B. Tapez ^C. Tapez les lettres PIP après A>. L'ordinateur répondra par une astérisque (*) sur la ligne après A>. Retirez la disquette du lecteur A. Placez la disquette de la base de données dans le lecteur A. Tapez B :*.* V . Cela aura pour effet de copier le contenu intégral et de vérifier le contenu de la disquette du lecteur A vers la disquette du lecteur B. Retirez la disquette de la base et la copie de sauvegarde. Replacez le disque original A dans le lecteur A. Replacez le disque original de la base de données dans le lecteur B. Appuyez sur la touche RETURN. L'ensemble de ce processus apparaît sur l'Ecran 6-9.

```
.QUIT

*** FIN DE SESSION      DBASE II ***

          ................ (retirez le disque contenant la base de B)
          ................ (insérez le disque de copie dans B)

A> ^C

A>PIP

          ................ (retirez le disque CP/M de A)
          ................ (insérez le disque contenant la base dans A)

*B:=*.* V
*

          ................ (retirez le disque contenant la base de A)
          ................ (insérez le disque CP/M dans A)

A> ^C

A>◆

          .... (vous avez maintenant un disque de sauvegarde dans B)
```

Ecran 6-9

Vous pourriez choisir la méthode consistant à faire deux copies de sauvegarde de vos fichiers. Les deux copies sont utilisées alternativement un jour sur l'autre. Cette méthode est profitable à long terme. Si vous la suivez strictement, vous risquez au maximum de perdre une journée de travail. Si vous pouvez limiter les risques à ce minimum (à l'exception des désastres : feu, etc.), vous avez fait tout ce que vous deviez faire. Il est actuellement possible de se protéger contre des incidents majeurs avec plus de facilité qu'avec des enregistrements papier. Il ne serait pas raisonnable de mettre à jour deux jeux d'enregistrements papier dans des endroits différents. On peut l'envisager avec une base de données sur ordinateur. Puisqu'il est possible de copier la base de données sur des disques souples ou sur bande magnétique, la ou les copie(s) peuvent facilement être rangées dans un endroit éloigné, protégé des incidents majeurs.

CHAPITRE VII

UTILISEZ VOTRE BASE DE DONNEES

Il y a réellement deux phases dans l'utilisation de votre base de données. La première, conserver des informations et les mettre à jour, a été traitée dans les chapitres précédents. La deuxième, obtenir qu'elle travaille pour vous, est le sujet de ce chapitre.



Deux utilisations fondamentales de la base de données

Il y a deux utilisations fondamentales d'une base de données.

1. Exécuter un certain nombre de services classiques, par exemple, la génération de fiches de paie, des états de T.V.A., la gestion de stocks, etc.

2. Obtenir des informations particulières lorsque cela est nécessaire.

Notre système de gestion de base de données, dBASE II, possède un éditeur de format d'impression organisé pour produire une grande variété de documents d'impression standard. Vous pouvez également écrire des « procédures » qui vous permettent de produire des états imprimés spécialisés et sur mesure correspondant à vos propres besoins. Une information particulière qui n'est pas habituelle peut être obtenue directement à partir du clavier de l'ordinateur en utilisant un « langage d'interrogation ». Nous discuterons d'abord des « états d'impression », que nous avons vus rapidement au Chapitre II, puis une étude des processus et des langages d'interrogation.

Etat imprimé (report)

Au Chapitre I, nous avons utilisé l'exemple de stock d'une boutique d'alcools pour s'initier au concept d'écriture d'un document imprimé. La plupart des systèmes de base de données modernes fournissent la possibilité de générer facilement des documents imprimés à partir du contenu d'une base de données. On peut directement, à partir du clavier, extraire des informations de la base pour les placer dans un format d'impression. Au Chapitre II, nous avons appris qu'un simple dialogue avec l'ordinateur met en place le processus de sortie d'un document dont l'ordinateur peut garder la trace du format. Nous pouvons utiliser ce format d'impression autant de fois que l'on veut. Ce processus est appelé par :

```
.REPORT
```

L'ensemble du dialogue avec l'ordinateur est reproduit à partir du Chapitre I par l'Ecran 7-1. Dans cet exemple, des commentaires supplémentaires ont été insérés en lettres minuscules. Le format d'impression que prépare l'utilisateur est établi à partir de la base de données B :STOCK, avec les réponses aux questions apparaissant sur l'Ecran 7-1. Pour obtenir le résultat imprimé de ce document, on tape la commande :

```
.REPORT TO PRINT
```

```
DONNEZ LE NOM DU FICHIER DE GENERATION D'ETAT : B:STOCK
ENTREZ LES OPTIONS, M=MARGE GAUCHE, L=LIGNES/PAGE, W=LARGEUR DE PAGE
EN-TETE DE PAGE ? (Y/N) Y
ENTREZ L'EN-TETE DE PAGE : INVENTAIRE D'UNE BOUTIQUE D'ALCOOLS
DESIREZ-VOUS UN DOUBLE INTERLIGNE ? (Y/N) N
DESIREZ-VOUS DES TOTAUX ? (Y/N) Y
DESIREZ-VOUS DES SOUS-TOTAUX DANS VOTRE RAPPORT ? (Y/N) Y
DONNEZ LE CHAMP DE SOUS-TOTAL : ALCOOL (sous-totaux pour chaque alcool)
DESIREZ-VOUS UNIQUEMENT UN RESUME DU RAPPORT ? (Y/N) N   (permet les
CHANGEMENT DE PAGE APRES LES SOUS-TOTAUX ? (Y/N) N        écritures
ENTREZ LE TITRE POUR LES SOUS-TOTAUX :                   individuelles)
COL     TAILLE,CONTENU
001     20,MARQUE                      (nb. espaces, nom de rubrique)
ENTREZ L'EN-TETE : MARQUE
002     10,CONT
ENTREZ L'EN-TETE : CONTENANCE
003     3,QUANTITE
ENTREZ L'EN-TETE : QTE
DESIREZ-VOUS DES SOUS-TOTAUX ? (Y/N) Y
004     6,PXRV
ENTREZ L'EN-TETE : PX REVIENT
DESIREZ-VOUS DES TOTAUX ? (Y/N) N  (rappelez-vous * signifie multiplié)
005     7,PXRV*QUANTITE        (prix de revient multiplié par quantité)
ENTREZ L'EN-TETE : VALORISATION
DESIREZ-VOUS DES TOTAUX ? (Y/N) Y
006
```

Ecran 7-1

PAGE NO.00001
15/01/86

STOCK D'UNE BOUTIQUE D'ALCOOLS

MARQUE		QTE	PXRV	VALORISATION
SCOTCH				
LONG JACK	90 cl	23	39.00	897.00
LONG JACK	1 l	7	43.00	301.00
LONG JACK	70 cl	88	29.80	262.24
** SOUS-TOTAL **		118		1.460.24
VODKA				
PETROVNA	2 l	35	23.78	832.30
PETROVNA	1 l	9	27.95	251.55
PETROVNA	90 cl	75	21.49	1.611.75
** SOUS-TOTAL **		119		2.695.60
WHISKEY				
4 ROSES	1 l 1/2	32	25.11	803.52
4 ROSES	2 l	44	31.98	1.407.12
4 ROSES	90 cl	19	18.50	351.50
NEW SOUTH	1/2 l	4	12.80	51.20
** SOUS-TOTAL **		99		2.613.12
BOURBON				
SPECIAL	70 cl	5	41.78	208.90
SPECIAL	90 cl	22	43.50	957.00
SPECIAL	1 l	21	58.10	1.220.10
SPECIAL	1 l 1/2	3	67.30	201.90
SPECIAL	3 l	5	80.80	404.00
** SOUS-TOTAL **		56		2.991.90
** TOTAL **		392		9.760.86

Figure 7-1. Sortie de document imprimé sur le stock d'une boutique d'alcools

Ce document fonctionne bien car les « types » d'alcools sont groupés ensemble dans la base de données. S'ils n'étaient pas groupés, ce serait le désordre. Si nous ne sommes pas sûr si les alcools sont regroupés ou non, nous devons d'abord indexer la base de données pour regrouper les données (de façon à assurer la cohérence du document).

```
.INDEX ON ALCOOL TO B:ALCOOL
.USE B:STOCK INDEX B:ALCOOL
```

Il y a un message dans la fonction report auquel nous avons répondu « non » à l'Ecran 7-1 qui est « DESIREZ-VOUS UNIQUEMENT UN RESUME DU RAPPORT (O/N) ? ». Si nous avions répondu « oui » à ce message, nous aurions obtenu un document différent tel qu'il est montré à la Figure 7-2. Vous pouvez remarquer qu'il n'y a pas d'informations dans la colonne prix de revient. Les seules valeurs imprimées sont les sous-totaux par catégorie.

```
PAGE NO.00001
15/01/86

                    STOCK D'UNE BOUTIQUE D'ALCOOLS

   MARQUE                                QTE   PX REVIENT  VALORISATION

*  SCOTCH                                118                  1.460.24

*  VODKA                                 119                  2.695.60

*  WHISKEY                                99                  2.613.12

*  BOURBON                                56                  2.991.90

** TOTAL **                             392                  9.760.86
```

Figure 7-2. Résumé du document préparé par l'ordinateur sur le stock d'alcools

Le manque d'informations dans la colonne Prix de revient résulte de la réponse « non » au message « DESIREZ-VOUS DES SOUS-TOTAUX DANS VOTRE RAPPORT ? ».

En utilisant l'exemple original, nous demandons un état imprimé sur le Bourbon :

```
.REPORT FOR ALCOOL='BOURBON'
```

Ce qui produit le rapport de la Figure 7-3. Ce document particulier ne fournit que des informations sur le Bourbon. La base de données étant très petite, le résultat n'a qu'une valeur d'illustration. Elle montre la possibilité d'édition de documents imprimés sur n'importe quel élément identifiable de la base de données. Cette possibilité devient réellement considérable si la base de données est importante. Dans ce cas, vous pouvez demander simplement l'impression d'un récapitulatif. Ces documents, comme le montre la Figure 7-3, sont édités à partir d'informations de synthèse.

```
PAGE NO.00001
15/01/85

                    STOCK D'UNE BOUTIQUE D'ALCOOLS

  MARQUE                                QTE   PX REVIENT   VALORISATION

*  BOURBON
SPECIAL                 70 CL              5       41.78         208.90
SPECIAL                 90 CL             22       43.50         957.00
SPECIAL                  1 L              21       58.10       1.220.10
SPECIAL                  1 L 1/2           3       67.30         201.90
SPECIAL                  3 L               5       80.80         404.00

** SOUS-TOTAL **                          56                   2.991.90

** TOTAL **                              392                   9.760.86
```

Figure 7-3. Document préparé par l'ordinateur pour tous les conditionnements de Bourbon extraits du stock d'une boutique d'alcools

Si nous voulons obtenir quelque chose de plus rapide, mais sans doute moins élégant, nous aurons recours à deux commandes de traitement du langage d'interrogation. Nous traiterons de ces langages d'interrogation et de leurs processus un peu plus tard, mais nous pouvons voir dès maintenant un exemple de comparaison des résultats de deux moyens principaux permettant d'obtenir des informations de votre base de données. Le processus de commandes et de réponses est représenté par l'Ecran 7-4.

```
.DISPLAY OFF MARQUE,QUANTITE,PXRV,PXRV*QUANTITE FOR
ALCOOL='BOURBON'

SPECIAL              70 CL         5      41.78        208.90
SPECIAL              90 CL        22      43.50        957.00
SPECIAL               1 L         21      58.10      1.120.10
SPECIAL               1 L 1/2      3      67.30        201.90
SPECIAL               3 L          5      80.80        404.00

.SUM QUANTITE,PXRV*QUANTITE

                                 392                 9.760.86
```

Ecran 7-4

Nous voyons donc à partir de cet exemple de stock d'une boutique d'alcools que l'extraction d'informations d'une base de données à partir d'un format d'impression est une opération très simple. Dans un environnement professionnel, il est facile et souhaitable de développer des documents d'impression adaptés à des cas particuliers à partir de votre base de données. Que ce soit une comptabilité clients, fournisseurs, une paie, un fichier du personnel, etc., ou un système de tenue de stock tel que celui-ci. Il est bon d'avoir une base de données constituant la source de toutes ces applications professionnelles. Chaque application séparée s'accommode facilement avec ses propres procédures de sortie d'états d'impression particulières. Et en même temps, toutes ces applications prennent leur origine dans une même base de données commune (un réservoir commun d'informations auquel on peut accéder pour des tâches non routinières en plus de la structure standard que vous avez mise en place et qui tourne à son propre rythme).

Un exemple. Supposons une société qui désire établir un contrat pour une nouvelle association commerciale. La direction aimerait connaître l'impact de ce contrat avant de le signer. Si, comme c'est souvent le cas même dans les gros systèmes de gestion, les applications de paie et de fichiers du personnel sont séparées, il faudra un effort important, si toutefois c'est possible, pour déterminer l'impact de cet éventuel contrat. Lorsqu'il y a de fait des systèmes séparés, on perd beaucoup de temps et d'énergie à dupliquer l'information et à la retrouver.

Par exemple, un système informatique qui établit les fiches de paie aura besoin du nom des employés, de leurs numéros, du nombre d'heures, du salaire horaire, etc. Un fichier du personnel aura besoin des mêmes informations. Lorsque les systèmes sont séparés, l'information qui est stocké par l'un ne peut pas toujours être utilisée par l'autre.

D'un autre côté, si le fichier du personnel et le système de paie utilisent un système de base de données, l'information peut être directement disponible à partir du clavier par une simple interrogation.

L'isolement d'informations nécessairement reliées est une des choses qui ont nécessité le développement des bases de données. Dans une base de données, tel qu'un « programme de comptabilité », ou pour tout ce qui est informatique, l'information stockée est indépendante de l'application. C'est à l'utilisateur d'imposer l'application qu'il souhaite voir développer lorsqu'il demande l'utilisation d'une base de données. Ceci nous ramène à un concept que nous venons de voir récemment : l'information disponible par un langage d'interrogation.

Extraire de l'information (QUERY)

Les systèmes de base de données répondent aux demandes d'information des utilisateurs (interrogations). L'élément du système de base de données qui traite de ce problème est appelé le QUERY LANGUAGE PROCESSOR (QLP), le Processeur de Langage d'Interrogation. Dans dBASE II, ce processeur est appelé le langage de développement des applications (ADL).

Vous utilisez le LANGAGE D'INTERROGATION pour dire à l'ordinateur ce qu'il faut faire. La plupart des langages d'interrogation contemporains ressemblent beaucoup à l'anglais courant. Dans certains systèmes, la seule fonction du langage d'interrogation sera d'extraire de l'information de la base de données. Dans ce cas, on dit que c'est une fonction à 'SIMPLE LECTURE'. Avec dBASE II, l'ADL est aussi utilisé pour METTRE A JOUR la base de données, ceci veut dire qu'il peut aussi bien « LIRE » qu'« ECRIRE ».

Les langages d'interrogation

Il y a deux sortes de langages d'interrogation :

procédural et non-procédural.

- Le langage procédural est traditionnel en informatique. Avec ce langage, vous dites à l'ordinateur, pas à pas, ce que vous voulez qu'il fasse pour produire vos résultats. Les exemples de langages procéduraux sont le BASIC, le FORTRAN, le PL/1 et le COBOL. Dans tous ces langages, vous dites à l'ordinateur comment trouver une réponse, et non pas « quel est le problème ».

- Le langage non-procédural vous permet de décrire le problème et l'ordinateur vous indiquera comment obtenir la réponse.

A partir de notre exemple d'annuaire téléphonique, si nous voulons savoir combien de personnes habitent Bordeaux,

```
.COUNT FOR 'BORDEAUX'$ADRESSE
```

Voilà un exemple de commande non-procédurale. Nous avons dit à l'ordinateur ce que nous voulons et il nous indique comment y parvenir.

Certains langages d'interrogation ont les options des deux langages. L'ADL de dBASE II est à la fois un langage d'interrogation procédural et non-procédural. Les options procédurales et non-procédurales des langages d'interrogation des systèmes de base de données relationnels sont quelquefois représentées dans les termes techniques suivants :

- Les options procédurales sont appelées l'ALGEBRE RELATIONNELLE.

- Les options non-procédurales sont appelées le CALCUL RELATIONNEL.

Il est clair que ces termes ont pour simple et unique intérêt d'essayer de faire peur à un non-professionnel.

Le besoin d'une information spécifique et inhabituelle est obtenue à partir du clavier de l'ordinateur en utilisant le langage d'interrogation. Si la base de données a beaucoup de rubriques, vous pourriez constituer et vous servir d'un dictionnaire de données (Chapitre III). Dans tous les cas, vous devez connaître les NOMS DE RUBRIQUES et ce qui est contenu dans les RUBRIQUES.

Un Langage d'interrogation qui vous permet un très haut niveau de demande à partir du clavier comprend trois parties dans sa syntaxe de commande. Ce sont

- le NOM,
- l'ETENDUE, et
- la CONDITION.

La commande 'NOM' est généralement représentative de la fonction que l'on attend d'elle. Les exemples dBASE II de noms de commandes sont DISPLAY, SUM, COUNT, LOCATE et LIST. L'étendue détermine sur quelle partie de la base de données s'applique la commande.

La condition signifie que la commande s'applique si l'enregistrement de la base remplit la condition qui est fixée. Les exemples de requête sont :

```
.SUM QUANTITE FOR ALCOOL='BOURBON'

.COUNT FOR 'Martin'$NOM

.DISPLAY QUANTITE FOR ALCOOL='SCOTCH' .AND. CONT='90 CL'

.DISPLAY QUANTITE,MARQUE FOR ALCOOL='SCOTCH' .AND. CONT='70 CL'
```

Dans le premier de ces quatre exemples, la commande signifie « Dites-nous combien vous avez de bouteilles de Bourbon ». Elle signifie exactement « Cumulez le contenu de la rubrique QUANTITE » lorsque le contenu de la rubrique ALCOOL est « BOURBON ». Si nous oublions le mot QUANTITE, cette commande n'aura pas de signification. Il n'est pas possible de donner à l'ordinateur des commandes vagues. Si nous le faisons, le résultat sera sans signification. Dans un bon langage d'interrogation, de telles tentatives seront rejetées par l'ordinateur.

Dans le deuxième exemple, la commande signifie « Comptez les enregistrements qui contiennent le nom Martin ». Le troisième exemple demande d'afficher la quantité (uniquement) pour chaque type de scotch dans la contenance 90 cl. Dans le quatrième exemple, on affiche en plus le nom de la marque.

On peut observer que la plupart de ces commandes utilisent un anglais courant. Ceci vient en partie du fait que nous avons utilisé pour chaque rubrique un nom particulièrement descriptif.

Avec l'ADL, l'enregistrement entier est affiché par la commande DISPLAY. Si vous n'en désirez pas tant, le fait d'entrer une liste de noms de rubriques séparés par des virgules (comme dans le quatrième exemple) indiquera à l'ordinateur d'afficher uniquement les rubriques mentionnées.

Le quatrième exemple utilise le mot AND d'une manière un peu particulière. Nous avons en fait dit à l'ordinateur (si le contenu de la rubrique ALCOOL est « SCOTCH » ET le contenu de la rubrique CONT est « 70 CL »), affichez la marque et sa quantité. Cette demande ira extraire exactement ce que nous cherchons.

Supposez que nous tapions cette demande :

```
.DISPLAY FOR ALCOOL='SCOTCH' .AND. ALCOOL='BOURBON'
```

Nous souhaitons l'affichage de tous les enregistrements dont le type d'alcool est SCOTCH et BOURBON. La réponse de l'ordinateur à cette commande est un point. Il n'y a pas d'affichage. Ce qui veut dire qu'il n'a rien trouvé. Comment cela est-il possible ? Nous savons qu'il existe des articles Bourbon et Scotch et l'ordinateur nous dit qu'il n'y en a aucun. Cette réponse « aucun » a tout de même un sens : il n'y a pas d'enregistrement dont le type d'alcool est à la fois Scotch et Bourbon. Si nous ré-écrivons la phrase comme suit : « Nous voulons l'affichage de tous les enregistrements dont le type d'alcool est soit Scotch soit Bourbon ». — nous avons la solution. Lorsque nous demandons l'information, nous pensons à l'ensemble de la base de données. L'ordinateur travaille avec un seul enregistrement à la fois.

Nous retapons la commande comme suit :

```
.DISPLAY FOR ALCOOL='SCOTCH' .OR. ALCOOL='BOURBON'
```

et nous obtiendrons le résultat souhaité.

Opérateurs BOOLEENS

Le ET et OU sont appelés des opérateurs BOOLEENS (AND et OR). Les opérateurs BOOLEENS sont souvent appelés des opérateurs logiques. Comme nous l'avons vu, ET et OU sont sensiblement équivalents aux ET et OU de la conversation courante. Lorsque nous utilisons ces « opérateurs », nous les écrivons avec un point à chaque extrémité pour les distinguer de la conversation courante. Nous devons être très attentif afin d'éviter les résultats étranges. Il n'est donc pas inutile de vous assurer que vous comprenez bien les opérateurs logiques. Dans le cas contraire, vous obtiendrez une réponse TECHNIQUEMENT correcte mais FAUSSE. Dans cette situation, on voit que l'ordinateur fait exactement ce qu'on lui dit de faire (et non pas ce que vous vouliez qu'il fasse). Un autre opérateur booléen utilisé avec dBASE II est .NOT. qui sera décrit plus loin au Chapitre X.

Quelques mots encore sur les interrogations

Les interrogations sont appropriées pour une grande variété de besoins de la part de l'utilisateur. Vous pouvez souhaiter, pour une raison ou pour une autre, d'avoir à manipuler l'information contenue dans une base de données, sans modifier les données de la base « officielle ». Nous avons déjà vu un exemple de ce type, lorsque nous avons « copié » une base de données pour pouvoir en modifier la structure. Une base de données peut être « copiée » en partie, sur une nouvelle base de données, pour un usage quelconque.

Faisons une copie de la partie de la base B:STOCK qui contient uniquement les rubriques MARQUE, CONT et PXVT. Pour restreindre un peu cette nouvelle base de données, nous copierons seulement les articles « scotch ». Cette opération s'effectuera par la commande.

```
.COPY FIELD MARQUE,CONT,PXVT TO B:SCOTCH FOR ALCOOL='SCOTCH'
```

Cela a pour effet de créer une base de données qui contient seulement du Scotch. Pour parler en termes plus techniques, cette opération de copie est appelée « la projection de la relation B:STOCK sur B:SCOTCH dans les limites du prédicat ALCOOL='SCOTCH' ». Cette description pompeuse vous convient-elle, reconnaissez-vous une simple opération de copie ?

Réfléchissons maintenant à un autre type de besoin. Par exemple, nous brûlons de savoir quel est l'article « alcool » dans le magasin, qui est le plus répandu et quel est l'article en moins grande quantité. Nous pouvons y parvenir en triant ou en indexant. Un index fournit une liste des enregistrements classés par quantité.

```
.INDEX ON QUANTITE TO B:QTE
00015 ENREGISTREMENT(S) INDEXE(S)
.USE B:STOCK INDEX B:QTE
```

Pour se placer au début de la base de données indexée, utilisons la commande dBASE II GO TOP. Pour se placer à la fin de la base de données, utilisons GO BOTTOM.

```
.GO TOP
.DISPLAY OFF MARQUE,CONT,QUANTITE
SPECIAL            1 L 1/2          3
.GO BOTTOM
.DISPLAY OFF MARQUE,CONT,QUANTITE
LONG JACK          70 CL           88
```

A partir du clavier, nous avons un moyen direct de réponse à notre question. Les articles les moins répandus sont les bouteilles de Bourbon Spécial d'un litre et demi et les articles les plus répandus sont les bouteilles de Scotch Long Jack de 70 cl.

Ces exemples d'interrogation donnent un petit aperçu de demandes d'interrogation de votre base de données. Les possibilités sont vraiment illimitées. Le plus important est de bien comprendre le principe et son fonctionnement pour aboutir à des interrogations qui soient intelligentes et qui s'appliquent à vos besoins.

Votre bloc-notes électronique

Vous avez accès à un autre type d'activité, potentiellement très utile, une sorte d'accessoire vous permettant de réaliser toutes sortes de choses avec votre base de données d'ordinateur.

L'information d'une simple rubrique dans un enregistrement peut être transférée dans un espace spécial de la mémoire centrale de l'ordinateur. Vous pouvez également obtenir que l'ordinateur stocke ce qui est frappé au clavier dans sa mémoire centrale. C'est un peu comme si vous disposiez d'un « bloc-notes » électronique.

Un système de base de données d'ordinateur fournit cette possibilité par l'intermédiaire de son processeur de langage d'interrogation. La possibilité de stocker temporairement l'information est bien commode lorsque l'on exécute des opérations manuelles au clavier. C'est indispensable lorsque l'on écrit des procédures pour automatiser des traitements.

Ce bloc-notes fonctionne un peu comme le système mémoire d'un calculateur électronique. Avec dBASE II, cela vous permet de :

- stocker 64 éléments séparés dans ce bloc-notes mémoire à tout moment
- emmagasiner pour chaque élément un maximum de 254 octets (caractères ou positions numériques)
- pouvoir compter sur une capacité maximale de 1 536 octets pour les 64 éléments.

Chaque élément stocké en mémoire est une VARIABLE MEMOIRE. Lorsqu'un élément (VARIABLE MEMOIRE) est stocké dans la mémoire, on doit lui fournir un nom. Le fait de nommer la variable mémoire permet de l'utiliser de manière commode et de conserver sa trace. Chaque système utilise un mot-clé pour indiquer à l'ordinateur d'accepter un élément dans une variable mémoire. Avec dBASE II, ce mot-clé est STORE.

Pour illustrer le fonctionnement de ce bloc-notes, nous stockerons le nombre 6 dans une variable mémoire que nous appellerons EXEMPLE.

```
.STORE 6 TO EXEMPLE
```

Rappelez-vous :

- Vous pouvez stocker en mémoire 64 éléments séparés à tout moment.
- Chaque fois que vous stockez un élément, vous lui donnez un nom.
- Si vous affectez le même nom à deux éléments différents, seul le deuxième élément sera enregistré.
- Pour modifier le contenu d'une variable mémoire, rangez simplement le nouvel élément dans l'ancien nom, par exemple, .STORE 7 TO EXEMPLE mettra la valeur 7 dans EXEMPLE à la place de 6 qui était son ancien contenu.
- Nous ne pouvons pas avoir deux éléments ou deux valeurs ayant le même nom au même moment.

Les variables mémoire peuvent être des chaînes de caractères, des nombres ou des informations logiques.

```
STORE 'ALPHABET' TO POTAGE

STORE T TO REPONSE
```

Il est pratiquement possible d'utiliser la mémoire pour n'importe quel usage. Vous pouvez combiner des chaînes de caractères dans une phrase :

```
STORE 'POTAGE' TO A

STORE 'ALPHABET' TO B

STORE A+B TO POTAGE
```

Pour voir le contenu d'une variable mémoire après le point, on utilise le point d'interrogation suivi du nom de l'élément.

```
.?POTAGE
  POTAGE ALPHABET
```

Les variables mémoire peuvent être utilisées pour exécuter des calculs arithmétiques.

```
.STORE 6 TO X
6
.STORE 7 TO Y
7
.STORE X+Y TO Z
13
.STORE Y-X TO W
1
.STORE X*Y TO Z        (Rappelez-vous * signifie multiplié)
42
.STORE X/Y TO D        (La barre oblique / signifie la division)
0
```

(nous avons ici un exemple de la perversité des ordinateurs. Pour obtenir l'affichage des positions décimales, nous devons indiquer à l'ordinateur combien nous en voulons.)

```
.STORE X*1.0/Y TO D
0.8
.STORE X*1.000/Y TO D
  0.857                    (vous vous souviendrez ?)
```

Pour avoir une meilleure idée de l'utilité du bloc-notes mémoire, examinons un livre de comptes bancaire informatique. Nous entrons les informations de rubrique suivantes : NUMERO, ORDRE, MONTANT, DEBIT et CREDIT. Pour déterminer la balance du compte, nous saisissons au clavier les écritures suivantes :

```
.SUM MONTANT FOR CREDIT TO CRED
17434.53
.SUM MONTANT FOR .NOT. CREDIT TO DEBIT
15997.18
.STORE CRED-DEBIT TO BALANCE
1437.35
```

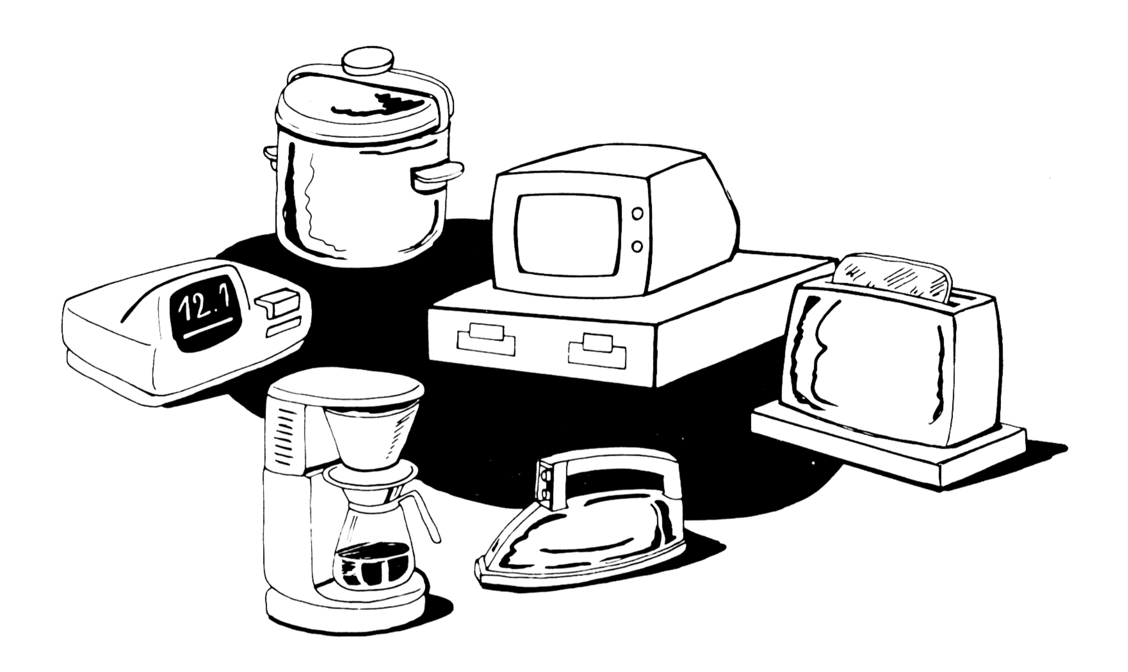
Nous venons d'obtenir un résultat très précieux, le solde du compte, directement à partir du clavier avec trois demandes. Pour comparer le résultat de la banque avec celui calculé par l'ordinateur lors de la réception des relevés de comptes mensuels, nous aurons :

```
.SUM MONTANT FOR CREDIT .AND. DEBIT TO CREDBANQ
16621.21
.SUM MONTANT FOR .NOT. CREDIT .AND. DEBIT TO DEB
15793.23
.STORE CREDBANQ-DEBIT TO BALBANQ
827.98
```

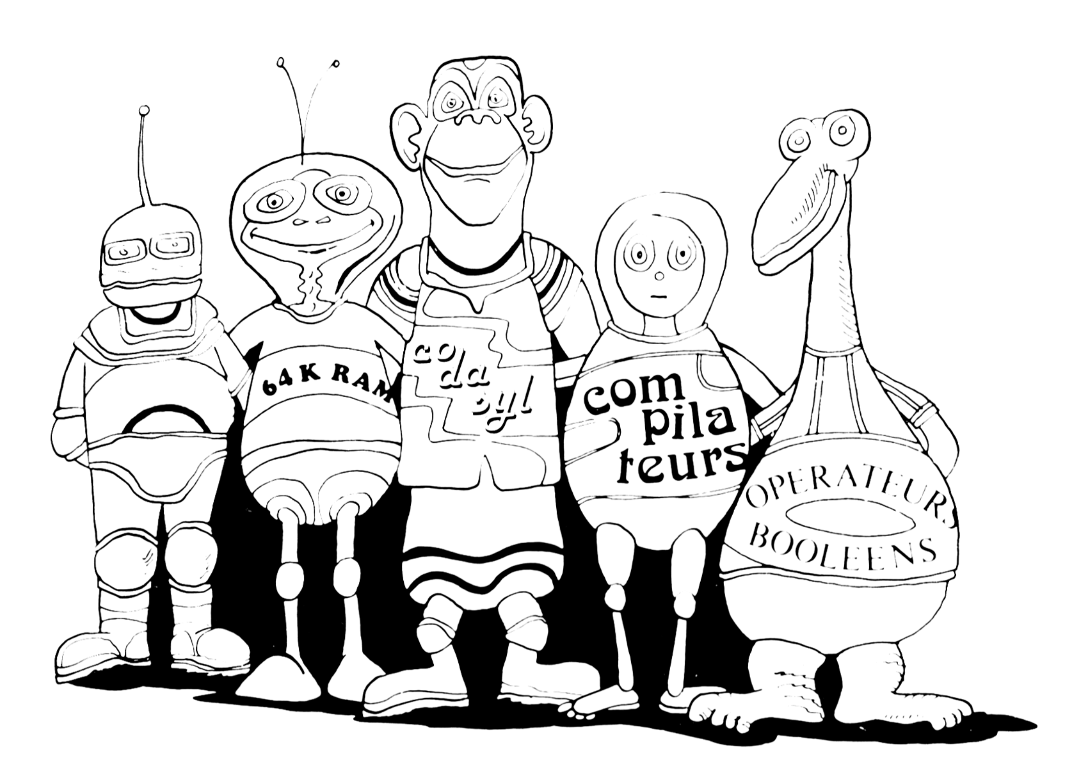
Nous avons donc obtenu, avec seulement trois instructions, un résultat très utile : la comparaison des résultats du relevé de la banque avec notre solde.

Dans ces deux exemples, nous avons obtenu les sommes que nous demandions sans avoir à les stocker. Nous aurions pu calculer le résultat avec un papier et un crayon ou une calculette. Nous risquions de faire une erreur en copiant les nombres à la main. Le « bloc-notes » mémoire nous fournit la possibilité d'utiliser l'ordinateur directement pour exécuter des opérations à partir des informations de la base de données. Si ces données sont correctes, il n'y a pratiquement pas de risque d'erreur.

Si maintenant vous avez le sentiment que tout cela est vraiment trop facile, que nous avons oublié quelque chose d'essentiel, soyez rassuré, la suite est également très facile.



La plupart de ce qui s'écrit actuellement sur les ordinateurs contribue en partie à montrer que l'informatique est quelque chose de difficile. Les textes s'adressent souvent aux lecteurs à partir de références issues d'un milieu technique. Dans un tel ouvrage, la simple explication de la commande .OR. serait considérablement différente de notre propos. Par exemple, la réponse de l'ordinateur à des exemples de commandes utilisant le OU logique serait décrite dans ces ouvrages comme « la relation B :STOCK dans les limites du prédicat ALCOOL='SCOTCH' » .OR. ALCOOL='BOURBON' ». Un prédicat, si vous voulez le savoir, représente le « rapport entre les valeurs et les propriétés ».



Il y a des fois où l'on se demande si les professionnels n'utilisent pas un « jargon » spécialisé pour justifier leur statut et leur salaire. C'est certainement inexact. Ce jargon permet aux spécialistes de mieux communiquer entre eux.

Mais, le langage technique pour les personnes qui n'appartiennent pas au « milieu informatique » et qui pensent donc que les ordinateurs sont étranges, inaccessibles, difficiles, etc. C'est pourquoi énormément de personnes se méfient lorsque l'on dit « c'est vraiment très facile ». En réalité, la plus grande part de difficulté vient de ce que l'on ne connaît pas. Les ordinateurs ne sont pas seulement pour les « professionnels de l'informatique ». Ils ont des possibilités incroyables qui sont facilement accessibles à toute personne qui doit stocker et diffuser de l'information.

Les bases de données stockent des données. Un SGBD vous permet d'extraire des informations à partir des données. Si nous avons une boutique d'alcools, nous pouvons apprendre aisément que nous sommes en rupture de stock de Téquila et surchargé en Scotch. Si nous combinons l'exemple d'une base de données Stock avec une base de données qui enregistre les livraisons des fournisseurs, nous pouvons

déterminer les ventes annuelles de chaque type et contenance d'alcools. Dans un temps déterminé, nous pouvons apprendre à gérer efficacement le stock. Ceci va permettre en retour une meilleure gestion financière et évidemment, une meilleure rentabilité de notre investissement. Il existe bien évidemment beaucoup d'autres applications à partir d'un système de gestion de base de données dans les affaires, l'éducation et l'administration. Le point important c'est la facilité avec laquelle on peut développer et utiliser efficacement un tel outil. Il peut nous aider dans toutes sortes d'entreprises. Cela peut être plaisant, et vous donner envie d'entreprendre toutes sortes d'activités intéressantes. Il peut vous permettre d'avoir une vision plus large d'une affaire à partir d'une seule et unique source d'informations.

TROISIEME PARTIE

La troisième partie traite de différents types de bases de données, en examinant la nature de chaque système et en dégageant leurs différences. Au Chapitre X, nous traiterons de la logique de l'ordinateur qui peut apparaître comme une notion compliquée au premier abord.

Le vocabulaire technique complexe peut vous donner l'impression que les ordinateurs sont des choses mystérieuses et/ou difficiles. Cette impression est inexacte et trompeuse — les ordinateurs sont vraiment des choses que l'on peut comprendre et très attractives. La troisième partie essaie de combler ces lacunes et répond à la plupart des questions que les gens se posent sur les ordinateurs. Ceci va permettre d'accroître votre compréhension du monde de l'informatique et vous amener à progresser avec vos bases de données d'ordinateur.

CHAPITRE VIII

LES CHOSES QUE VOUS VOUDRIEZ SAVOIR

Il y a un certain nombre de questions que l'on pose souvent au sujet des systèmes de gestion de base de données micro-informatiques. Dans ce chapitre et dans les suivants, nous essaierons de répondre à la plupart de ces questions. Qu'est-ce qu'un système en langage d'assemblage ? Qu'est-ce qu'une base de données hiérarchique ? Qu'est-ce qu'une base de données en réseau ? Qu'est-ce que le CODASYL ? Que sont..... ? ? ?

Le LANGAGE D'ASSEMBLAGE est le « langage natif » de l'ordinateur. Souvent appelé par erreur le « langage machine », c'est le « langage » que l'ordinateur utilise de façon interne. Au-dessus des langages d'assemblage, on trouve les « langages évolués » tels que FORTRAN, COBOL, PASCAL et le LANGAGE DE DEVELOPPEMENT D'APPLICATION de dBASE II (ADL). Chaque commande ou instruction de ces langages est construite à partir de nombreuses instructions en langage d'assemblage.

Lorsque l'ordinateur exécute une commande telle que DISPLAY, il exécute en réalité un grand nombre d'instructions en langage d'assemblage. Les langages de haut niveau tels que le BASIC et ADL utilisent une moyenne de 50 à 100 instructions en langage d'assemblage pour chaque instruction en langage évolué. Un système de gestion de base de données en langage d'assemblage est fait d'une construction d'une multitude d'instructions en langage d'assemblage.

Beaucoup de packages logiciels du commerce sont écrits en langage évolué tel que le PASCAL ou le BASIC. La raison en est qu'il est plus facile d'écrire des logiciels en langage évolué et aussi que ces logiciels sont plus « portables ». La portabilité signifie que le même logiciel peut être aisément adapté et utilisé sur toutes sortes de micro-ordinateurs.

Un programme en langage d'assemblage s'utilise uniquement sur un type de machine. Pour qu'il puisse être utilisé sur différents types d'ordinateurs, il doit être ré-écrit de façon particulière pour chaque type d'ordinateur.

Par contre, il est généralement admis que les programmes en langage d'assemblage rendent la machine, sur laquelle ils sont implantés, beaucoup plus performante dans l'utilisation de ses ressources qu'un programme en langage évolué tel que le BASIC. C'est probablement vrai si nous comparons un programme bien écrit en langage d'assemblage avec un programme bien écrit en langage évolué.

Comme il est très difficile de juger si deux systèmes sont bien écrits, on ne doit pas prendre en compte cet élément lorsque l'on choisit un produit du commerce. Chaque démarche dans le choix d'un système logiciel se justifie. A partir du moment où vous ne verrez jamais de publicité commerciale pour un système déclarant que le produit est « médiocre » ou « moyennement bon », vous ne pourrez jamais intégrer ce critère dans votre sélection. Le plus important pour vous c'est : fera-t-il votre travail ?

L'accès séquentiel et aléatoire

L'ACCES SEQUENTIEL ET L'ACCES ALEATOIRE se retrouvent souvent dans les articles sur les bases de données et quelquefois dans les publicités. Ces termes s'appliquent à la manière dont l'ordinateur obtient les informations. L'accès séquentiel signifie que l'ordinateur démarre du premier enregistrement et parcourt l'ensemble du fichier de base de données en séquence jusqu'à ce qu'il trouve l'enregistrement que vous désirez. Les commandes en langage d'interrogation telles que DISPLAY et LOCATE utilisent une démarche séquentielle.

```
.DISPLAY FOR NOM='MARTIN, GEORGES'

.LOCATE FOR NOM='MARTIN, GEORGES'
```

Lorsque l'on utilise DISPLAY de cette manière, l'ordinateur examine chaque enregistrement de la base de données — commençant par l'enregistrement 1 — dans l'ordre de leur numéro d'enregistrement (séquentiellement). Lorsque l'on utilise LOCATE, l'ordinateur examinera chaque enregistrement de la base — commençant par l'enregistrement 1 et procédant séquentiellement jusqu'à ce qu'il rencontre un enregistrement contenant MARTIN, GEORGES dans la rubrique NOM. C'est un accès séquentiel pour un enregistrement particulier. Notez que le temps mis par l'ordinateur pour trouver un enregistrement particulier dépend de la position de l'enregistrement dans la base. Si la base de données est importante, et que l'enregistrement se trouve près de la fin, il faudra plusieurs secondes avant que l'ordinateur puisse trouver cet enregistrement. Si l'ordinateur peut « lire » l'ensemble de la base de données en une minute, le temps moyen d'accès sera de 30 secondes.

Le terme ACCES ALEATOIRE est quelque peu incorrect. Il ne signifie pas que l'ordinateur cherche au hasard dans la base de données jusqu'à ce qu'il trouve l'enregistrement souhaité. Cela veut dire explicitement que l'ordinateur accède directement à chaque enregistrement de la base.

Le terme « aléatoire » signifie que si vous choisissez n'importe quel enregistrement aléatoirement, l'ordinateur obtiendra cet enregistrement aussi rapidement qu'un autre.

L'exemple trivial d'une base de données papier conçue pour un accès direct est l'annuaire. L'annuaire est imprimé par ordre alphabétique. Lorsque vous cherchez le numéro de téléphone d'une personne particulière, vous utilisez la nature alphabétique de l'annuaire pour trouver le nom de la personne et récupérer le numéro que vous souhaitez. Pour pouvoir obtenir le numéro par un accès séquentiel, il vous faudrait commencer par le premier nom de l'annuaire et parcourir celui-ci jusqu'à ce que vous trouviez le nom que vous cherchez. L'accès séquentiel est simple, sûr et relativement lent. L'accès direct (aléatoire) est une méthode plus rapide pour obtenir un enregistrement particulier.

Les clés primaires et secondaires

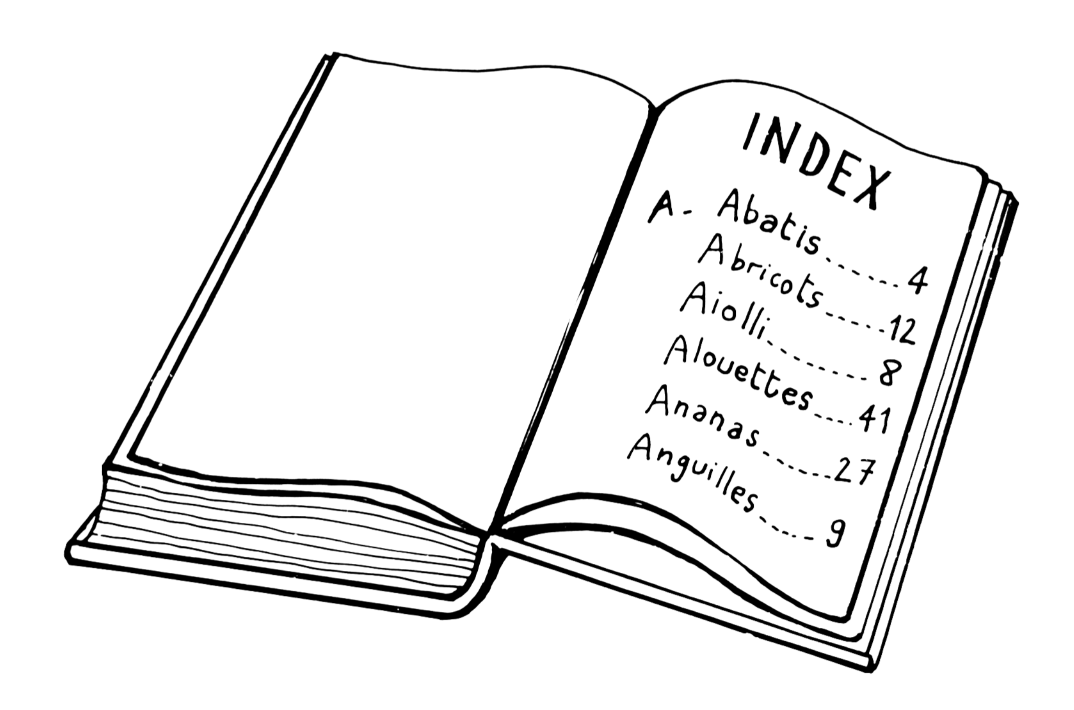
Si l'on veut que l'ordinateur accède directement à un enregistrement particulier, il faut l'aider. Cette aide est matérialisée par une « CLE PRIMAIRE », attachée au sommet de l'enregistrement. L'ordinateur utilise cette CLE pour aller directement vers cet enregistrement, soit par calcul soit en retrouvant l'information de position de l'enregistrement sur le disque. Par conséquent, lorsque l'enregistrement est demandé, l'ordinateur va directement sur la position physique de l'enregistrement et la recopie en mémoire centrale. L'information « clé » est ajoutée à l'ordinateur lors de la création de l'enregistrement. Avec dBASE II, le NUMERO D'ENREGISTREMENT représente la « clé primaire » ajoutée par l'ordinateur. Si vous connaissez le numéro d'enregistrement, vous pouvez accéder directement à celui-ci. Repérer un enregistrement par sa « clé » est très rapide, mais impose des restrictions puisque l'utilisateur doit d'abord connaître la « CLE ». Avec dBASE II, connaître la « clé » signifie connaître le NUMERO D'ENREGISTREMENT.

La clé primaire doit être unique. Il n'est pas envisageable que vous puissiez vous rappeler de la clé primaire de chaque enregistrement dans une base de données d'ordinateur. On peut l'envisager si la base de données n'est pas très importante, cependant si cette base est réduite, l'accès direct n'est plus nécessaire.

Une CLE SECONDAIRE est une solution au problème d'utilisation efficace d'une clé primaire. Une base de données peut avoir une ou plusieurs clés secondaires. Les clés utilisées par l'ordinateur ne sont généralement pas d'un très grand usage pour nous et les « clés » significatives pour nous ne sont généralement pas très utiles pour l'ordinateur qui essaie d'obtenir un accès direct. Les clés secondaires nous fournissent ce lien nécessaire.

Les clés secondaires ne fournissent pas réellement un accès direct, mais s'en rapprochent. L'accès est plus rapide que le temps moyen d'accès en séquentiel. Les tables fournissent la « conversion » entre les clés primaires et secondaires.

Un livre de cuisine ordinaire donne un bon exemple quotidien de clés primaires et secondaires.



La clé primaire est le numéro de la page. Les clés secondaires sont le nom des plats et des ingrédients. L'index est la table qui « convertit » le nom de l'ingrédient en numéro de page. Le numéro de page vous amène à l'élément que vous souhaitez. Si vous savez que la recette recherchée est à la page 123, c'est certainement plus simple que d'utiliser l'index. Cependant, en utilisant l'index, cela va plus vite que de parcourir l'ensemble du livre de cuisine. Vous pouvez rencontrer plus d'un numéro de page pour le même élément. Ces éléments sont des exemples de clés non uniques. Les clés secondaires n'ont pas besoin d'être uniques.

Supposons une base de données Elèves. Chaque enregistrement possède le nom, l'adresse, le numéro de téléphone, la classe, le niveau, etc. Nous aimerions être capable d'accéder aux enregistrements des élèves par le nom ou le niveau, ou le numéro de la classe. Pour y parvenir, nous identifierons ces trois rubriques comme des clés secondaires. Nous aurons trois tables, une pour chaque clé. Quand nous voudrons un enregistrement d'élève particulier, nous utiliserons le nom de la clé et nous demanderons l'élève par son nom. L'ordinateur trouvera le nom dans la table, récupérera le numéro d'enregistrement et utilisera ce dernier pour obtenir l'enregistrement de la base de données.

Puisque les tables sont destinées à un usage particulier, on pourra les construire pour accéder directement aux éléments de données que l'on recherche. L'accès direct avec dBASE II est fourni au moyen de tables que l'on appelle des fichiers d'index. Le système construira les tables avec la commande INDEX. Pour construire ces tables, on procède comme suit :

```
.USE B:ECOLE
.INDEX ON NOM TO B:NOM
.INDEX ON CLASSE TO B:CLASSE
.INDEX ON NIVEAU TO B:NIVEAU
```

Ces commandes viennent de construire trois tables permettant d'utiliser les trois rubriques NOM, CLASSE et NIVEAU comme des clés secondaires. Ces trois tables ont des noms de fichiers B:NOM, B:CLASSE et B:NIVEAU. Le système de base de données ajoute .NDX à la fin du nom de fichier pour indiquer à l'ordinateur que c'est un FICHIER D'INDEX, et servira ensuite à accéder à un enregistrement par la clé secondaire. Si vous souhaitez trouver des enregistrements s'accordant avec le nom de l'élève

```
.USE B:ECOLE INDEX B:NOM
```

Vous pouvez maintenant accéder directement à chaque enregistrement d'élève en utilisant uniquement le nom de l'élève avec la commande FIND.

```
.FIND Portnoy, Rémy
```

Incidemment, si Portnoy est le seul élève de l'école dont le nom commence par Po, vous pourriez utiliser

```
.FIND Po
```

ce qui aboutirait au même résultat. L'ordinateur vient de trouver l'enregistrement de Portnoy. Pour visualiser l'enregistrement, tapez simplement le mot DISPLAY. Si vous êtes intéressé par la 6ème, vous pourriez taper :

```
.USE B:ECOLE INDEX B:NIVEAU
.FIND 6
```

Ces deux commandes vous placeront sur le premier enregistrement d'un élève de niveau 6ème. Tous les élèves de niveau 6ème seront groupés ensemble dans l'ordre de leur numéro d'enregistrement. Puisqu'ils sont tous regroupés, nous pouvons afficher tous les élèves de 6ème par la commande

```
.DISPLAY WHILE NIVEAU='6'
```

Lorsque l'on utilise une base de données avec un fichier d'index (en utilisant des clés secondaires), l'exécution de l'ordinateur pour une commande telle que DISPLAY FOR NIVEAU='6' sera sensiblement plus longue lorsque l'on utilisera cette base de données sans le fichier d'index. Les fichiers d'index, puisqu'ils peuvent être utilisés pour « trier » les enregistrements par groupe, permettront de se passer de commandes qui réorganisent physiquement la base de données — telles que SORT. Non seulement l'opération d'indexation n'affecte en rien l'organisation physique de la base mais elle est beaucoup plus rapide que le processus de tri (SORT).

Nous pouvons également indexer plus d'une rubrique en même temps. Par exemple, supposons que nous voulions obtenir des élèves groupés par classe (classe et niveau) et que nous les voulions par ordre alphabétique dans leur classe. Nous utiliserons les trois rubriques accolées ensemble comme une clé secondaire. Comme exemple :

```
.USE B:ECOLE
.INDEX ON NIVEAU+CLASSE+NOM TO B:CLASSE
```

Le signe « + » signifie « lier les trois rubriques ensemble pour former une seule clé ». Si nous voulions afficher l'ensemble de la base de données par la commande DISPLAY ALL, les élèves apparaîtraient alors dans l'ordre de niveau par classe à l'intérieur du niveau et par ordre alphabétique à l'intérieur d'une classe. L'ordinateur conservera ces fichiers sur disque de telle façon qu'ils puissent être utilisés une fois encore sans avoir à ré-indexer. La ré-indexation peut prendre beaucoup de temps, particulièrement si la clé a plusieurs caractères et si la base de données est très importante. La ré-indexation est uniquement nécessaire si l'on ajoute ou supprime des enregistrements ou si l'on modifie la rubrique clé.

La plupart des systèmes de gestion de base de données fournissent une ré-indexation « automatique ». Avec dBASE II, cette ré-indexation se fait lorsque l'on ajoute, modifie ou efface les enregistrements, avec les fichiers d'index. Dans notre exemple,

```
.USE B:ECOLE INDEX B:CLASSE,B:NOM,B:NIVEAU
```

mettra à jour les fichiers index B:NOM, B:CLASSE AND B:NIVEAU chaque fois que l'on ajoutera un enregistrement, que l'on modifiera une rubrique « clé » ou que l'on effacera des enregistrements avec les commandes DELETE et PACK.

Cette méthode possède quelques inconvénients. Mettre à jour de multiples fichiers d'index (multiples clés secondaires) est une opération très lente. Cela n'est pas unique à dBASE II. La mise à jour de multiples tables de clés secondaires dans n'importe quel système de gestion de base de données prend du temps. Pour cette raison, il est fortement recommandé de s'abstenir d'utiliser des clés multiples (multiples fichiers d'index). Un autre inconvénient viendra si une modification est faite sur la base de données, sans avoir intégré la modification sur le fichier d'index. Cela peut désorganiser la table ou n'importe quel traitement informatique exécuté à l'aide de cette table.

Les tables de clés secondaires (les fichiers d'index), bien entendu, nécessitent un certain espace sur le disque. Si vous disposez de plus d'un lecteur de disque, vous pouvez placer la table d'index sur un lecteur de disque séparé de la base de données. Les tables n'ont pas besoin d'être placées sur le même disque que la base. Cependant, ces lecteurs de disque doivent être en ligne. Ceci veut dire que l'ordinateur doit avoir un accès simultané à la base de données et au fichier d'index que vous utilisez. Un fichier d'index (comme une base de données) doit être contenu entièrement sur une seule disquette comme dans la plupart des systèmes de base de données micro-informatiques. Cet élément doit être pris en compte lorsque vous concevez votre système de base de données et lorsque vous choisissez le matériel informatique sur lequel vous voulez implanter votre système.

Les enregistrements physiques et logiques

Au Chapitre I, nous avons pris l'exemple de gestion d'une boutique de pièces détachées automobiles comme analogie pour expliquer le concept d'un système de gestion de base de données. Comme nous l'avons vu, l'employé, ses catalogues de pièces détachées, et les rayons de son magasin, sont assimilables à un SGBD tandis que les pièces détachées présentes sont analogues aux informations qui sont stockées dans la base de données. Si vous étudiez un peu les systèmes de base de données par des ouvrages, vous trouverez des références aux ENREGISTREMENTS PHYSIQUES et aux ENREGISTREMENTS LOGIQUES. Les enregistrements physiques sont les actuels enregistrements de données — ils correspondent aux pièces détachées dans l'exemple ci-dessus. Les enregistrements logiques sont les écritures dans les tables de clés secondaires (fichiers d'index) qui indiquent à l'ordinateur où se trouvent les enregistrements physiques sur le disque. Ces enregistrements correspondent aux éléments des catalogues qu'utilise l'employé pour connaître l'emplacement des pièces détachées automobiles.

Au Chapitre I, nous avons également utilisé une bibliothèque comme analogie d'un système de gestion de base de données. Dans l'exemple de la bibliothèque, les enregistrements physiques correspondent aux livres. Les enregistrements logiques correspondent aux fiches du catalogue de la bibliothèque.

Dans notre exemple de bibliothèque, il existe quelque chose d'autre ayant un rapport avec les enregistrements logiques, et servant à garder la trace des livres. Ce sont les fiches de sortie des livres. Vous avez dû les voir lorsqu'à la bibliothèque, un livre que vous souhaitiez emprunter est sorti. La bibliothécaire peut consulter ses enregistrements pour vous dire la date à laquelle le livre sera rendu. Si vous le souhaitez, le livre peut vous être réservé et vous serez avisé lors de sa rentrée. Cela peut vous faire gagner beaucoup de temps.

Enregistrements verrouillés

Comme dans notre exemple de bibliothèque, plusieurs personnes peuvent avoir besoin d'utiliser un enregistrement physique au même moment. Il n'est pas réellement souhaitable que plus d'une personne puisse accéder à un enregistrement de la base en même temps.

Pour illustrer ce point, supposons que la base de données contienne les réservations de places d'un vol aérien. Si deux employés de la compagnie d'aviation veulent

utiliser l'information sur le vol en même temps, ils risquent de réserver le même siège sur ce vol, à deux clients différents. Pour éviter ce genre de chose, le SGBD réserve l'enregistrement au premier demandeur. Un deuxième demandeur sera informé que l'enregistrement est en utilisation. Le SGBD « réservera » l'enregistrement pour le deuxième demandeur dès qu'il deviendra disponible. Tous les demandeurs de cet enregistrement ont une interdiction d'accès jusqu'à ce qu'il soit abandonné par le premier utilisateur. Le terme technique de cette situation est le VERROUILLAGE DE L'ENREGISTREMENT.

Il est pour l'instant inopérant pour un système de gestion de base de données micro-informatique de fournir la possibilité de verrouiller l'accès à un enregistrement. La plupart des systèmes micro-informatiques courants sont conçus pour être utilisés par une seule personne à la fois — il s'agit d'ordinateurs personnels —. Il se dessine cependant une tendance au développement de micro-ordinateurs partagés par plus d'une personne à la fois. Ce qui permet le partage de périphériques coûteux qui seraient autrement sous-utilisés, tels qu'une imprimante. Comme les systèmes de gestion de base de données deviennent plus répandus sur les systèmes micro-informatiques, ils contribueront certainement au développement d'ordinateurs multi-utilisateurs. Ces systèmes sont organisés de manière à pouvoir « partager » l'information. La possibilité de partager les informations rapidement et économiquement a une très grande valeur. C'est la principale raison du développement des systèmes de gestion de base de données sur les ordinateurs de grosse configuration. C'est également la raison du développement de systèmes matériels abordables de gestion partagée d'information pour les petites entreprises.

CHAPITRE IX

LES CHOSES QUE VOUS VOUDRIEZ SAVOIR *(SUITE)*

Il existe trois types de systèmes de base de données :

- RELATIONNEL
- HIERARCHIQUE
- EN RESEAU.

Leur différence vient de la façon dont est organisée l'information. Dans un système relationnel, nous envisageons l'information sous forme de tableaux, avec des lignes et des colonnes. Un système hiérarchique ressemble à l'organigramme hiérarchique d'une société. Un système en réseau peut s'imaginer comme les organigrammes de deux sociétés qui auraient fusionné. La plupart des nouveaux systèmes de gestion de base de données sont soit RELATIONNELS soit une version des systèmes en RESEAUX appelée CODASYL.

La base de données relationnelles

Ce livre se concentre principalement sur le système de base de données RELATIONNEL car il correspond davantage à l'expérience quotidienne. La plupart des systèmes de gestion de base de données micro-informatiques sont des variations de l'idée RELATIONNELLE. Le système de base de données relationnel a le même état physique que la présentation des données. L'information est gérée et stockée comme le font la plupart des gens, d'une manière naturelle. Si l'on utilise un système relationnel, une personne non familière et inexpérimentée avec les systèmes d'ordinateur peut obtenir des résultats intéressants avec une relative facilité.

Un exemple de base de données relationnelle simple (extraite du stock de notre boutique d'alcools du Chapitre II) est présenté à la Figure 9-1.

ALCOOL	MARQUE	CONT	QUANTITE	PXRV	RXVT
SCOTCH	LONG JACK	90 CL	32	39.00	45.70
SCOTCH	LONG JACK	1 L	7	43.00	58.30
SCOTCH	LONG JACK	70 CL	88	29.80	40.10
VODKA	PETROVNA	2 L	35	23.78	33.50
VODKA	PETROVNA	1 L	9	27.95	37.90
VODKA	PETROVNA	90 CL	75	21.49	31.30
WHISKEY	4 ROSES	1 L 1/2	32	25.11	40.10
WHISKEY	4 ROSES	2 L	44	31.98	41.00
WHISKEY	4 ROSES	90 CL	19	18.50	28.50
WHISKEY	NEW SOUTH	1/2 L	4	12.80	22.20
BOURBON	SPECIAL	70 CL	5	41.78	52.25
BOURBON	SPECIAL	90 CL	22	43.50	51.00
BOURBON	SPECIAL	1 L	21	58.10	67.15
BOURBON	SPECIAL	1 L 1/2	3	67.30	79.12
BOURBON	SPECIAL	3 L	5	80.80	91.50

Figure 9-1. Exemple d'une base de données relationnelle

Cette base de données relationnelle se présente exactement comme si nous avions établi le stock en utilisant un crayon et du papier. Chaque enregistrement a une longueur fixe. Chaque rubrique dans l'enregistrement a toujours la même taille.

La base de données hiérarchique

Les systèmes de base de données HIERARCHIQUES nécessitent que nous imaginions l'information organisée suivant une hiérarchie. Le diagramme d'une base de données hiérarchique ressemble à un organigramme de société. Il ressemble aussi à un arbre à l'envers. Les structures hiérarchiques sont souvent appelées des structures en arbre. La base de données de la boutique d'alcools peut être représentée par la Figure 9-2.

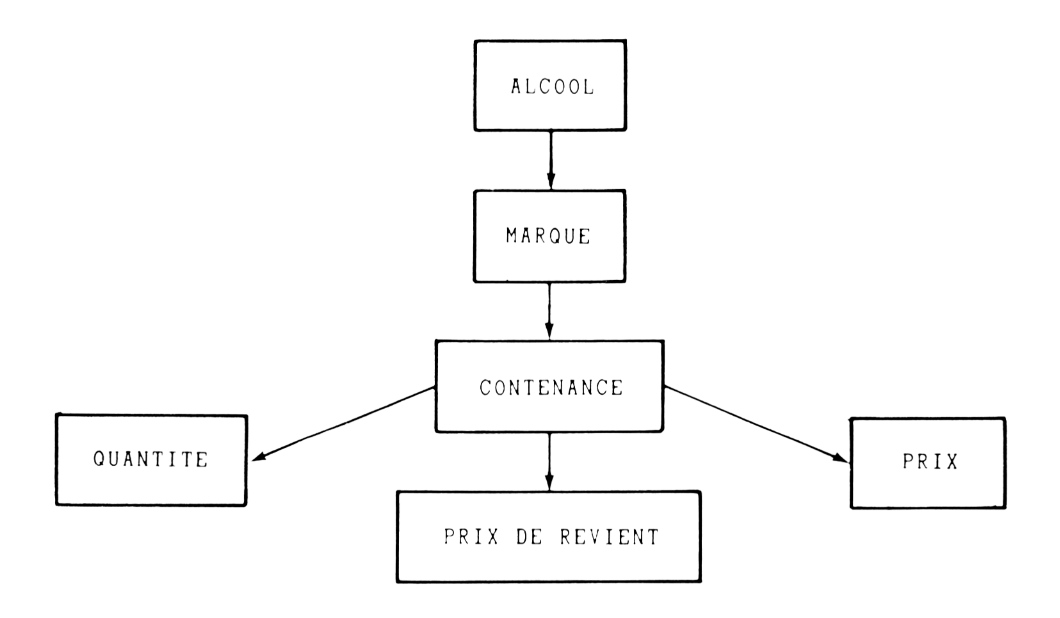


Figure 9-2. Représentation hiérarchique du stock d'une boutique d'alcools

Lorsqu'on la représente de cette manière, la structure hiérarchique devient apparente. Chaque élément de données est subordonné à un autre élément de données (excepté le premier bien sûr, celui du rectangle supérieur). L'« enregistrement » n'est plus une simple collection de rubriques mais c'est une collection de sous-enregistrements, ou « segments ». Chaque rectangle dans cet exemple est un « segment » ou un élément de l'enregistrement. Un segment peut contenir plus d'une rubrique. Les trois derniers rectangles, par exemple, peuvent être regroupés pour former un seul segment.

L'utilisation du mot enregistrement est inappropriée puisqu'il est difficile de séparer un type d'« enregistrement » d'un autre dans notre esprit. Dans cette version de notre base de données stock d'alcools, nous avons quatre « enregistrements » : un pour chacune des quatre différentes sortes d'alcools représentées dans la base de données relationnelle B :STOCK : (scotch, vodka, whiskey, bourbon). L'enregistrement hiérarchique « whiskey » correspond à la donnée « whiskey » de notre base de données relationnelle présentée à la Figure 9-3.

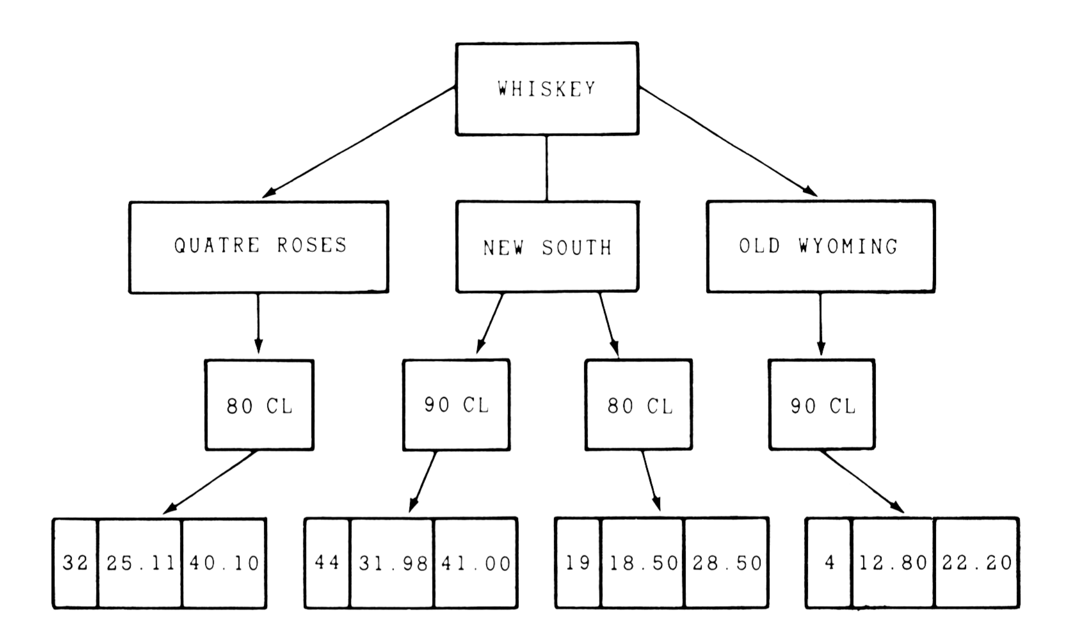


Figure 9-3. L'enregistrement hiérarchique "Whiskey"

Dans ces systèmes, chaque segment dépend d'un autre segment. Aucun segment ne peut dépendre de plus d'un segment. Cependant, un segment peut être propriétaire de plus d'un segment. Les propriétaires sont souvent appelés les parents. Les segments subordonnés, les enfants. Dans l'exemple de la Figure 9-3, Whiskey est le parent de Quatre roses, New South et Old Wyoming. Chacun de ces enfants, à son tour, est le parent (ou propriétaire) des différents segments CONTENANCE. Ils sont également les ENFANTS de « Whiskey ».

Voyons maintenant comment cela fonctionne. Chaque segment a un code qui lui est rattaché, ce code est unique pour chaque segment (clé primaire). Il identifie le type du segment et le numéro d'ordre dans la séquence (exactement comme un numéro d'enregistrement de dBASE II).

Chacun des segments de ALCOOL contient plusieurs pointeurs. Chaque pointeur conduit l'ordinateur sur un segment MARQUE qui lui est subordonné. Les segments MARQUE, à leur tour, contiennent des pointeurs qui conduisent l'ordinateur vers chacun des segments CONTENANCE issus de cette MARQUE. Chacun de ces segments,

à leur tour, contiennent des pointeurs qui conduisent l'ordinateur vers les segments contenant QUANTITE, PRIX DE REVIENT et PRIX DE VENTE. Ces segments peuvent ne contenir aucun pointeur. Les pointeurs permettent à l'ordinateur d'aller directement d'un segment à l'autre pour assembler un enregistrement complet.

Pourquoi cette complexité pour faire un simple travail. Dans un système de base de données relationnelle, nous n'avons pas ce genre de pointeurs pour regrouper chaque enregistrement. Si l'on reprend l'exemple relationnel (dBASE II), nous utilisons quatre enregistrements pour le stock (« Whiskey »). Dans la version Hiérarchique, nous n'utilisons plus qu'un segment « Whiskey » et quelques pointeurs. Si notre stock d'alcools a mille articles et qu'il se compose de seulement dix sortes d'alcools, nous éviterons beaucoup de duplication d'informations ou de redondance à l'aide d'un système hiérarchique.

Permettre d'éviter les redondances semble être une bonne idée. Quels en sont les inconvénients ? Le premiers de tous, la redondance (répétition), est réduite au dépend de la simplicité. Beaucoup d'applications peuvent aisément s'intégrer dans cette structure, certaines n'y parviennent pas. De plus, nous devons savoir à l'avance ce que sera notre application.

Le fait qu'un segment dépende d'un seul parent est une restriction évidente d'un système hiérarchique. Supposons que nous ayons une base de données hiérarchique du personnel. Deux de nos employés se marient et ont un enfant. Le segment d'enregistrement de la base de données pour cet enfant ne peut pas dépendre des enregistrements du personnel des deux parents à la fois. Bien entendu, ce critère est certainement dépassé à l'heure actuelle car beaucoup d'améliorations ont été apportées pour contrecarrer cet exemple. Néanmoins, il illustre bien l'inconvénient classique d'un système hiérarchique.

Les pages jaunes de l'annuaire d'une base de données papier fournissent un moyen de se représenter une base de données hiérarchique. Si nous représentons les pages jaunes comme il est montré à la Figure 9-4, la nature hiérarchique paraît évidente.

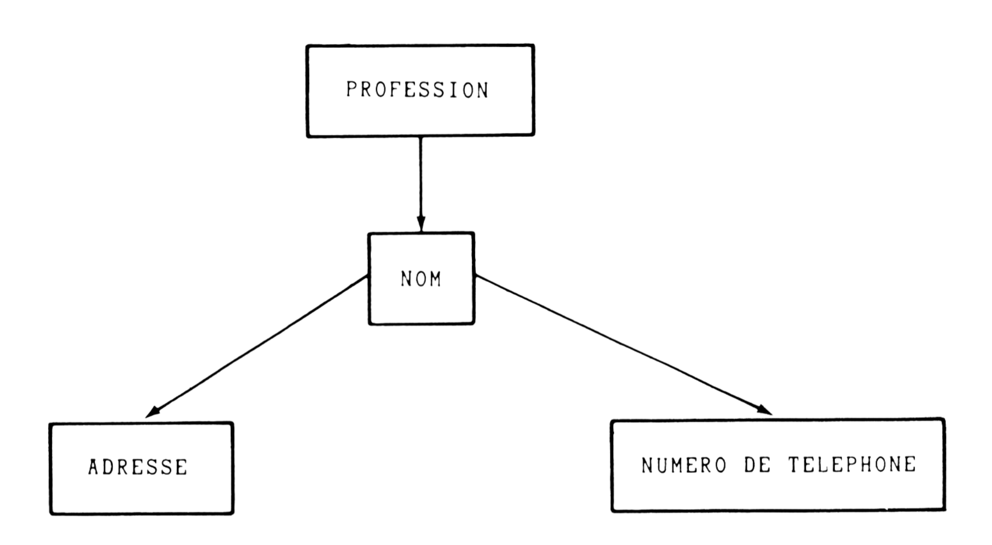


Figure 9-4. Représentation hiérarchique des pages jaunes

L'ADRESSE et le NUMERO DE TELEPHONE sont issus du NOM, le NOM, à son tour, dépend de la PROFESSION. D'une autre manière, PROFESSION est le propriétaire de NOM. NOM, à son tour, est le propriétaire à la fois de ADRESSE et NUMERO DE TELEPHONE. Un segment ne peut pas appartenir à plus d'un propriétaire. Un propriétaire peut cependant avoir la propriété de plusieurs autres segments.

Contrairement à une base de données relationnelle, les enregistrements peuvent être de tailles variées. Quelques enregistrements, par exemple, peuvent avoir plusieurs numéros de téléphone tandis que d'autres n'en ont qu'un seul. Cette situation peut être gérée par une base de données relationnelle mais elle oblige de « gâcher » de la mémoire ou d'être très vigilant. Pour les professions qui ont plus d'une adresse et/ou numéro de téléphone, la structure peut se représenter comme à la Figure 9-5.

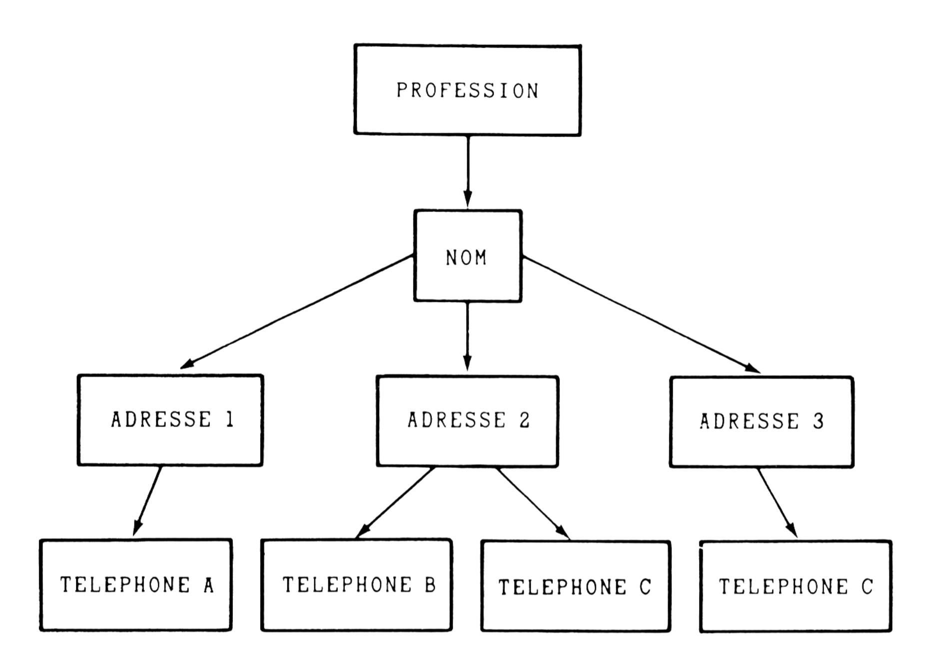


Figure 9-5

Pour avoir une meilleure idée, supposons que nous ayons un enregistrement représentant les vendeurs de voitures neuves. Le segment PROFESSION contient le titre « Vendeurs de Voitures Neuves » plus quelques pointeurs qui conduisent l'ordinateur vers tous les segments contenant le nom de ces vendeurs. L'un de ces vendeurs est « Vroom Vroom Motors ». Le segment pour « Vroom Vroom Motors » contient le nom de la société ainsi qu'un jeu de pointeurs conduisant l'ordinateur aux adresses des différents locaux de la société. Ce système de base de données peut utiliser le segment ID (identification) pour maintenir un ordre dans l'enregistrement (par exemple, mettre les noms de sociétés dans l'ordre alphabétique). Chaque segment adresse doit contenir l'adresse et les pointeurs des numéros de téléphone. Un segment contiendra toujours les pointeurs qui dirigeront l'ordinateur vers des segments inférieurs. Un segment POURRAIT contenir un pointeur dirigeant l'ordinateur vers son segment propriétaire.

Il y a un certain nombre d'inconvénients à un système de base de données hiérarchique, du point de vue de l'utilisateur. Premièrement, si nous voulons un listing alphabétique de tous les noms de la base, cela représente un certain effort. Deuxièmement, il est envisageable qu'une société puisse être répertoriée dans plus d'une catégorie professionnelle. Vroom Vroom Motors peut très bien être un atelier de réparation, un département de pièces détachées, et un service de voitures d'occasion en plus de ses attributions de distributeur de voitures neuves. Dans un système hiérarchique, un « enfant » ne peut avoir qu'un seul parent. Notre exemple de vendeur de voiture peut être listé sous trois catégories de professions. Pour cela, l'activité doit être saisie trois fois, une fois dans chaque catégorie.

La base de données en réseau

Les systèmes de gestion de base de données EN RESEAU ressemblent aux systèmes hiérarchiques. La différence majeure est que sous certaines conditions, un « enfant » peut avoir plus d'un « parent ». Une autre différence est que la relation « parent-enfant » peut être interrompue. Finalement, le vocabulaire est différent suivant que l'on a affaire à un système hiérarchique ou à un système relationnel.

Le terme RESEAU est souvent utilisé alternativement avec CODASYL. C'est parce que la plupart des systèmes de gestion de base de données en RESEAU courants sont basés sur la proposition de standard américain pour les bases de données. Ce standard a été développé par le Data Base Task Group (DBTG) de la Conférence sur les langages de systèmes de données (CODASYL) qui a développé le langage informatique COBOL. Le principe d'une base de données en réseau repose sur le concept de sets (les ensembles dans les mathématiques modernes). Les bases de données en réseau sont plus complexes que les systèmes hiérarchiques, elles offrent cependant une plus grande souplesse.

Dans un système de base de données en RESEAU, la base de données est constituée par une collection de SETS. Chaque set se compose d'une collection d'enregistrements. Un enregistrement ressemble à celui d'un système relationnel. Mais dont la longueur serait variable. Un enregistrement peut être issu de plus d'un set. Un set est un groupe d'éléments semblables. Il y a par exemple plusieurs NOMS de professions dans la catégorie PROFESSION. Par conséquent, tous les NOMS sont subordonnés au set profession. Dans l'annuaire, les vendeurs de voitures neuves tels que Vroom Vroom Motors sont issus du set des vendeurs de voitures neuves. Vroom Vroom Motors est appelé un membre du set des vendeurs de voitures neuves. L'enregistrement propriétaire est « Vendeurs de voitures neuves ». Chaque set doit avoir un

enregistrement propriétaire. Un set peut être composé d'un seul enregistrement. Un enregistrement ne peut pas dépendre de deux occurences du même type de set. Par conséquent, la situation illustrée ci-après n'est pas permise.

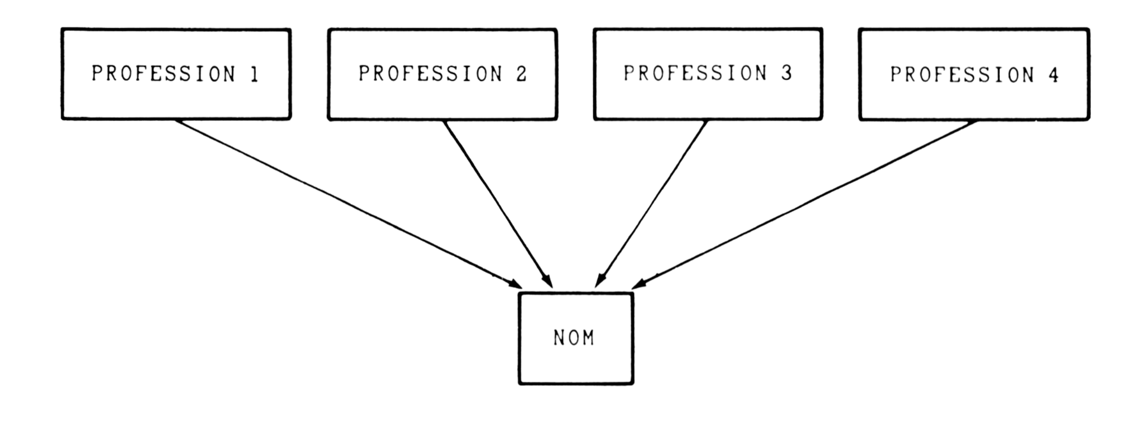


Figure 9-6

Maintenant, bien entendu, l'exemples des pages jaunes devient caduque car nous savons qu'un nom de société peut très bien être représenté sous plusieurs catégories de profession dans les pages jaunes. Nous avons là un exemple où le nombre possible de relations est énorme. Il peut devenir tellement important qu'il dépasserait la capacité des plus gros ordinateurs. La base de données en réseau est confortable lorsque l'on a affaire à une ou plusieurs relations. Il faut donc restructurer les sets de telle façon que toutes les relations soient du type un à plusieurs (1 :n).

Une caractéristique particulière d'un système de base de données hiérarchique vient du fait que si le NOM dépend de PROFESSION, un listing alphabétique de tous les noms sera difficile à obtenir. La raison en est que les NOMS sont mis en ordre alphabétique pour chaque catégorie de profession et leur accès ne peut se faire que par le type de profession. Dans un système en réseau, NOM peut dépendre de PROFESSION et, en même temps, la PROFESSION peut dépendre du NOM.

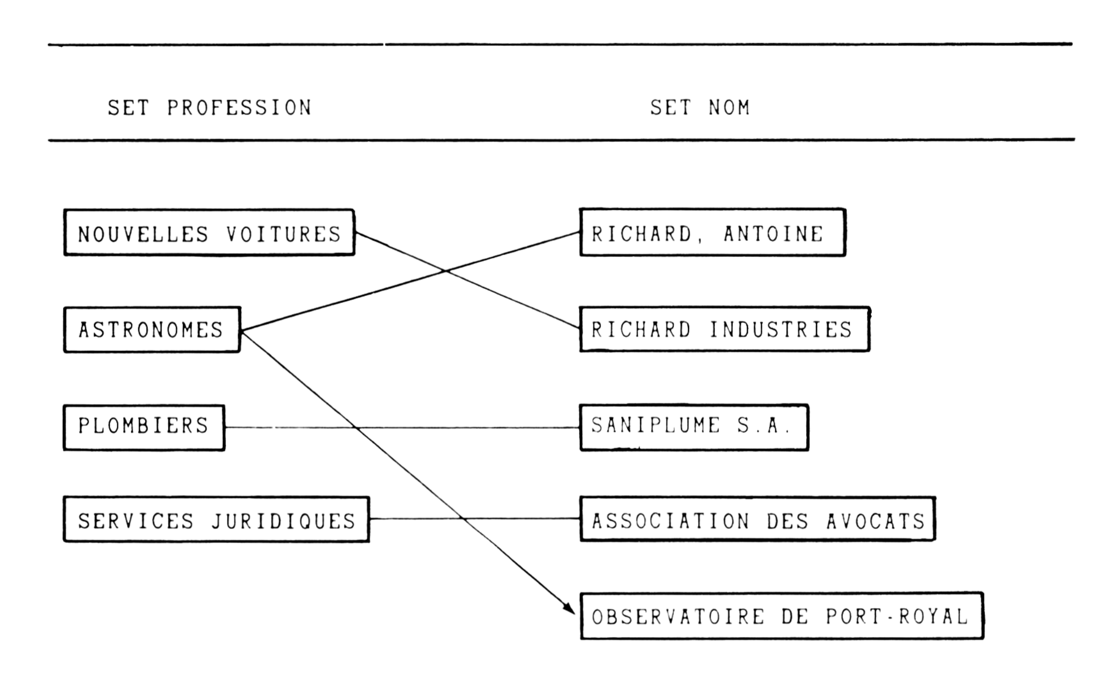


Figure 9-7

Comme nous pouvions le supposer, la base de données en réseau utilise une terminologie différente de celles des systèmes hiérarchiques ou relationnels. Le « data item » (élément de donnée) est similaire à ce que nous appelions une rubrique. Dans l'exemple de la base de données hiérarchique, il pouvait y avoir plusieurs numéros de téléphone « dépendant » d'une adresse. C'est ce que l'on appelle une « relation conceptuelle ». Il peut également y avoir plusieurs adresses dépendant d'un nom. Chacune des adresses a un numéro de téléphone. L'élément adresse-numéro de téléphone est appelé un groupe. S'il y a plus d'un groupe, c'est un groupe répété.

Les systèmes de base de données HIERARCHIQUE et EN RESEAU sont de structures fondamentalement différentes d'un système RELATIONNEL. La façon dont les enregistrements de données sont structurés dans ces systèmes est nettement plus complexe que dans les systèmes relationnels. Ces systèmes sont bien indiqués pour des applications de base de données complexes et de capacités importantes. Pour

cette raison, ils sont largement utilisés sur de gros ordinateurs. C'est la raison pour laquelle ils sont très performants dans l'exploitation de toutes les ressources de l'ordinateur, le temps machine du processeur et la mémoire centrale. Leur efficacité potentielle peut devenir extrêmement importante lorsque la base de données contient dix mille ou cent mille enregistrements. Comme le coût de mise en œuvre d'une configuration d'un gros système d'ordinateur peut facilement avoisiner plusieurs milliers de dollars de l'heure, la valeur de ces performances est toute relative. Le coût de travail de programmeurs professionnels avec ces systèmes de base de données est facilement justifié lorsque leur travail peut réduire le coût d'exploitation du système de gestion de base de données sur de gros ordinateurs. Il n'est pas évident que ce type de performance soit aussi indispensable avec des bases de données implantées sur des micro-ordinateurs.

Comme nous l'avons vu précédemment, chacun de ces trois systèmes peut fournir toutes les fonctions nécessaires à l'exploitation d'une base de données. Chacun possède ses points forts et ses points faibles. Les systèmes hiérarchiques/en réseau offrent à l'utilisateur l'efficacité et la vitesse. Ils permettent d'optimiser l'exploitation des ressources de l'ordinateur.

Ils sont cependant complexes et relativement lourds. Ils ont été développés pour un usage sur de grosses configurations d'ordinateurs où le stockage sur disque en ligne avoisine couramment le billion d'octets. Un billion d'octets représente de une à quatre mille disquettes souples 8 pouces. La lecture de cette énorme capacité de stockage à la vitesse de lecture d'un disque souple prendrait un demi-million de seconde. Ce qui représenterait des JOURS. Si nos besoins en base de données ont réellement cette importance, un système de base de données sur micro-ordinateur ne conviendra pas. Alors, il n'est pas inconcevable que vous ayez besoin d'un système de base de données hiérarchique ou en réseau et un gros organisateur.

Chacun de ces trois types de systèmes de base de données a ses avantages. Chacun a ses points forts et ses points faibles. Chacun peut exécuter n'importe quelle tâche sur la base de données. Les systèmes hiérarchiques ou en réseau nécessitent que vous vous représentiez l'information sous forme d'une hiérarchie ou d'un réseau. Cela implique que vous dessiniez au préalable le « dessin logique » de votre base de données pour permettre l'utilisation des données suivant un chemin particulier. Ce qui veut dire que lorsque vous « créez » la base de données, vous devez déjà savoir comment vous allez l'utiliser. La démarche relationnelle nécessite que vous vous représentiez l'information en termes de tableaux de lignes et de colonnes. La manière dont vous utiliserez les données pourra être déterminée plus tard.

CHAPITRE X

UN PEU DE LOGIQUE

Les ordinateurs et les systèmes de gestion de base de données sont construits à partir de systèmes logiques. La plupart des systèmes de gestion de base de données micro-informatiques sont conçus de telle façon que la logique puisse être un moyen d'expression naturel. L'utilisation de la logique n'est pas difficile — en fait, c'est même quelque chose de plaisant. Cependant, vous observerez que la compréhension de la logique de l'ordinateur vous permet d'aller plus loin avec votre ordinateur et votre SGBD.

Il existe trois termes logiques courants que l'on appelle des opérateurs.

.AND.
.OR.
.NOT.

Comme nous l'avons déjà vu dans ce livre, tout cela est très proche du langage naturel. Les points, à chaque extrémité du mot, font partie de l'opérateur logique.

Pour illustrer l'utilisation de ces termes, nous prendrons l'exemple du stock d'une boutique d'alcools du Chapitre II. La base de données est présentée à la Figure 10-1.

ALCOOL	MARQUE	CONT	QUANTITE	PXRV	PXVT
SCOTCH	LONG JACK	90 CL	23	39.00	45.70
SCOTCH	LONG JACK	1 L	7	43.00	58.30
SCOTCH	LONG JACK	70 CL	88	29.80	40.10
VODKA	PETROVNA	2 L	35	23.78	33.50
VODKA	PETROVNA	1 L	9	27.95	37.90
VODKA	PETROVNA	90 CL	75	21.49	31.30
WHISKEY	4 ROSES	1 L 1/2	32	25.11	40.10
WHISKEY	4 ROSES	2 L	44	31.98	41.00
WHISKEY	4 ROSES	90 CL	19	18.50	28.50
WHISKEY	NEW SOUTH	1/2 L	4	12.80	22.20
BOURBON	SPECIAL	70 CL	5	41.78	52.25
BOURBON	SPECIAL	90 CL	22	43.50	51.00
BOURBON	SPECIAL	1 L	21	58.10	67.15
BOURBON	SPECIAL	1 L 1/2	3	67.30	79.12
BOURBON	SPECIAL	3 L	5	80.80	91.50

Figure 10-1.

Supposons que nous souhaitons examiner toutes les entrées de WHISKEY et de BOURBON. Il nous est maintenant naturel d'écrire la commande d'interrogation suivante :

```
DISPLAY FOR ALCOOL='BOURBON'.AND.ALCOOL='WHISKEY'
```

Malheureusement, cela ne marche pas. Cette commande indique à l'ordinateur d'afficher tous les enregistrements où la rubrique ALCOOL contient à la fois WHISKEY et BOURBON. Il n'en existe aucune. La logique de l'ordinateur s'applique à un enregistrement à la fois, et non à toute la base de données. La commande correcte est :

```
DISPLAY FOR ALCOOL='BOURBON'.OR.ALCOOL='WHISKEY'
```

Cette commande s'applique à la partie de la base de données qui est montrée non hachurée à la Figure 10-2.

ALCOOL	MARQUE	CONT	QUANTITE	PXRV	PXVT
SCOTCH	LONG JACK	90 CL	32	39.00	46.70
SCOTCH	LONG JACK	1 L	7	43.00	58.30
SCOTCH	LONG JACK	70 CL	88	29.80	40.70
VODKA	PETROVNA	2 L	95	23.78	33.50
VODKA	PETROVNA	1 L	9	27.95	37.80
VODKA	PETROVNA	90 CL	75	21.49	31.30
WHISKEY	4 ROSES	1 L 1/2	32	25.11	40.10
WHISKEY	4 ROSES	2 L	44	31.98	41.00
WHISKEY	4 ROSES	90 CL	19	18.50	28.50
WHISKEY	NEW SOUTH	1/2 L	4	12.80	22.20
BOURBON	SPECIAL	70 CL	5	41.78	52.25
BOURBON	SPECIAL	90 CL	22	43.50	51.00
BOURBON	SPECIAL	1 L	21	58.10	67.15
BOURBON	SPECIAL	1 L 1/2	3	67.30	79.12
BOURBON	SPECIAL	3 L	5	80.80	91.50

Figure 10-2

L'emploi correct de l'opérateur .AND. (ET) retrouve la zone commune à deux groupes. Par exemple, pour déterminer quels sont les enregistrements qui contiennent whiskey et la contenance 90 cl, on tapera la commande suivante :

```
DISPLAY FOR ALCOOL='WHISKEY'.AND.CONT='90 CL'
```

L'ordre des rubriques n'a pas d'importance. Le même résultat est obtenu par :

```
DISPLAY FOR CONT='90 CL'.AND.ALCOOL='WHISKEY'
```

Si nous avions utilisé .OR. à la place de .AND. dans ce dernier exemple, nous aurions obtenu un résultat complètement différent.

```
DISPLAY FOR CONT='90 CL'.OR.ALCOOL='WHISKEY'
```

La partie non hachurée de la Figure 10-3 indique les enregistrements qui s'afficheraient comme résultat de cette commande.

ALCOOL	MARQUE	CONT	QUANTITE	PXRV	PXVT
SCOTCH	LONG JACK	90 CL	32	39.00	45.70
SCOTCH	LONG JACK	1 L	7	43.00	58.30
SCOTCH	LONG JACK	70 CL	88	29.80	40.10
VODKA	PETROVNA	2 L	35	23.78	33.50
VODKA	PETROVNA	1 L	9	27.95	37.90
VODKA	PETROVNA	90 CL	75	21.49	31.30
WHISKEY	4 ROSES	1 L 1/2	32	25.11	40.10
WHISKEY	4 ROSES	2 L	44	31.98	41.00
WHISKEY	4 ROSES	90 CL	19	18.50	28.50
WHISKEY	NEW SOUTH	1/2 L	4	12.80	22.20
BOURBON	SPECIAL	70 CL	5	41.78	52.25
BOURBON	SPECIAL	90 CL	22	43.50	51.00
BOURBON	SPECIAL	1 L	21	58.10	67.15
BOURBON	SPECIAL	1 L 1/2	8	67.30	79.12
BOURBON	SPECIAL	3 L	5	80.80	91.50

Figure 10-3

Supposons que nous voulons extraire les enregistrements de contenance 90 cl soit de Whiskey ou de Bourbon. Nous mettrons en place la commande suivante :

```
DISPLAY FOR (ALCOOL='WHISKEY'.OR.ALCOOL='BOURBON').AND.CONT='90 CL'
```

Les enregistrements qui répondent à ces critères sont montrés dans la zone non hachurée de la Figure 10-4.

ALCOOL	MARQUE	CONT	QUANTITE	PXRV	PXVT
SCOTCH	LONG JACK	90 CL	32	39.00	45.70
SCOTCH	LONG JACK	1 L	7	43.00	58.30
SCOTCH	LONG JACK	70 CL	88	29.80	40.10
VODKA	PETROVNA	2 L	35	23.78	33.50
VODKA	PETROVNA	1 L	9	27.95	37.90
VODKA	PETROVNA	90 CL	75	21.49	31.30
WHISKEY	4 ROSES	1 L 1/2	32	25.11	40.10
WHISKEY	4 ROSES	2 L	44	31.98	41.00
WHISKEY	4 ROSES	90 CL	12	18.50	28.50
WHISKEY	NEW SOUTH	1/2 L	4	12.80	22.20
BOURBON	SPECIAL	70 CL	5	41.78	52.25
BOURBON	SPECIAL	90 CL	88	43.50	51.00
BOURBON	SPECIAL	1 L	21	58.10	67.15
BOURBON	SPECIAL	1 L 1/2	3	67.30	79.12
BOURBON	SPECIAL	3 L	5	80.80	91.50

Figure 10-4

Supposons que nous ayons omis les parenthèses dans le dernier exemple. La commande se présente comme suit :

```
DISPLAY FOR ALCOOL='WHISKEY'.OR.ALCOOL='BOURBON'.AND.CONT='90 CL'
```

L'affichage résultant — qui est complètement différent — est représenté à la Figure 10-5. Voilà ce qui arrive lorsque l'opérateur .AND. (ET) a la suprématie sur l'opérateur .OR. (OU).

ALCOOL	MARQUE	CONT	QUANTITE	PXRV	PXVT
SCOTCH	LONG JACK	90 CL	32	39.00	45.70
SCOTCH	LONG JACK	1 L	7	43.00	56.30
SCOTCH	LONG JACK	70 CL	88	29.80	40.10
VODKA	PETROVNA	2 L	35	23.78	33.50
VODKA	PETROVNA	1 L	9	27.95	37.90
VODKA	PETROVNA	90 CL	75	21.49	31.30
WHISKEY	4 ROSES	1 L 1/2	32	25.11	40.10
WHISKEY	4 ROSES	2 L	44	31.98	41.00
WHISKEY	4 ROSES	90 CL	19	18.50	28.50
WHISKEY	NEW SOUTH	1/2 L	4	12.80	22.20
BOURBON	SPECIAL	70 CL	5	41.78	52.25
BOURBON	SPECIAL	90 CL	22	43.50	51.00
BOURBON	SPECIAL	1 L	21	56.10	67.15
BOURBON	SPECIAL	1 L 1/2	3	67.30	79.12
BOURBON	SPECIAL	3 L	5	80.80	91.50

Figure 10-5

L'arrangement logique résultant de la zone non hachurée de la Figure 10-4 est dû à la structure de la commande :

```
DISPLAY FOR (ALCOOL='WHISKEY'.OR.ALCOOL='BOURBON').AND.CONT='90 CL'
```

Maintenant supposons que nous voulions réellement quelque chose d'autre. Vous risquez de passer à côté si vous n'écrivez pas logiquement. Par contre, il sera peut-être plus facile d'obtenir ce que nous cherchons en utilisant l'opérateur .NOT. (NON). La commande se présente alors comme suit :

```
DISPLAY FOR .NOT.((ALCOOL='WHISKEY'.OR.ALCOOL='BOURBON').AND.
  .CONT='90 CL')
```

L'expression logique complète est placée entre parenthèses pour indiquer à l'ordinateur que le .NOT. s'applique à tous les éléments. Le résultat est présenté par les zones non hachurées de la Figure 10-6.

ALCOOL	MARQUE	CONT	QUANTITE	PXRV	PXVT
SCOTCH	LONG JACK	90 CL	32	39.00	45.70
SCOTCH	LONG JACK	1 L	7	43.00	58.30
SCOTCH	LONG JACK	70 CL	88	29.80	40.10
VODKA	PETROVNA	2 L	35	23.78	33.50
VODKA	PETROVNA	1 L	9	27.95	37.90
VODKA	PETROVNA	90 CL	75	21.49	31.30
WHISKEY	4 ROSES	1 L 1/2	32	25.11	40.10
WHISKEY	4 ROSES	2 L	44	31.98	41.00
WHISKEY	4 ROSES	90 CL	19	18.50	28.50
WHISKEY	NEW SOUTH	1/2 L	4	12.80	22.20
BOURBON	SPECIAL	70 CL	5	41.78	52.25
BOURBON	SPECIAL	90 CL	22	43.50	51.00
BOURBON	SPECIAL	1 L	21	58.10	67.15
BOURBON	SPECIAL	1 L 1/2	3	67.30	79.12
BOURBON	SPECIAL	3 L	5	80.80	91.50

Figure 10-6

Maintenant, supposons que nous ayons besoin de voir tous les enregistrements WHISKEY et BOURBON pour toutes les contenances excepté 90 cl. Nous obtiendrons ce résultat par :

```
DISPLAY FOR (ALCOOL='WHISKEY'.OR.ALCOOL='BOURBON').AND..NOT.
  .CONT='90 CL'
```

Les enregistrements concernés sont montrés par les zones non hachurées de la Figure 10-7.

ALCOOL	MARQUE	CONT	QUANTITE	PXRV	PXVT
SCOTCH	LONG JACK	90 CL	32	39.00	45.70
SCOTCH	LONG JACK	1 L	7	43.00	58.30
SCOTCH	LONG JACK	70 CL	88	29.80	40.10
VODKA	PETROVNA	2 L	35	23.78	33.50
VODKA	PETROVNA	1 L	9	27.95	37.90
VODKA	PETROVNA	90 CL	75	21.49	31.30
WHISKEY	4 ROSES	1 L 1/2	32	25.11	40.10
WHISKEY	4 ROSES	2 L	44	31.98	41.00
WHISKEY	4 ROSES	90 CL	19	15.50	28.50
WHISKEY	NEW SOUTH	1/2 L	4	12.80	22.20
BOURBON	SPECIAL	70 CL	5	41.78	52.25
BOURBON	SPECIAL	90 CL	22	43.50	51.00
BOURBON	SPECIAL	1 L	21	58.10	67.15
BOURBON	SPECIAL	1 L 1/2	3	67.30	79.12
BOURBON	SPECIAL	3 L	5	80.80	91.50

Figure 10-7

Vous pouvez vous aider des caractéristiques du langage pour vous éviter des problèmes de logique compliqués. Par exemple,

```
DISPLAY FOR ALCOOL$'BOURBON   ,WHISKEY'
```

donnera le même résultat que :

```
DISPLAY FOR ALCOOL='BOURBON'.OR.ALCOOL='WHISKEY'
```

La première ligne indique à l'ordinateur d'afficher chaque enregistrement qui correspond au contenu de la rubrique ALCOOL placé entre apostrophes. Les espaces blancs sont présents puisque l'ordinateur comparera l'ensemble de la rubrique ALCOOL avec la chaîne de caractères. La rubrique a dix caractères. Si vous omettiez les espaces, l'ordinateur ne trouverait pas une correspondance avec le contenu d'une rubrique. Les espaces blancs ont, pour l'ordinateur, autant de signification que n'importe quel autre caractère.

Si nous souhaitons examiner tous les enregistrements excepté BOURBON ou WHISKEY, la commande correcte est la suivante :

```
DISPLAY FOR .NOT.ALCOOL$'BOURBON          ,WHISKEY'
```

Le signe $ est un moyen abrégé qui signifie « contenu dans ». On l'appelle quelquefois un opérateur de chaîne. Les espaces supplémentaires sont présents puisque la rubrique ALCOOL a dix espaces (taille = 10). Si l'on n'avait pas mis ces espaces, l'ordinateur n'aurait pas été capable de trouver un équivalent dans cette rubrique avec cette chaîne de caractères. La virgule n'est pas nécessaire dans ce cas, cependant c'est une bonne habitude que de séparer les éléments qui peuvent correspondre avec des caractères ne faisant pas partie de la comparaison.

L'utilisation d'opérateurs logiques — .AND., .OR. et .NOT. — vous permet d'indiquer précisément à l'ordinateur à quelles conditions s'appliquent les commandes. Ils améliorent l'efficacité de recherche les enregistrements particuliers qui vous intéressent.

Un peu plus loin dans cet ouvrage, nous étudierons les procédures qui mettent en oeuvre une assistance de l'ordinateur. Les opérateurs logiques deviendront même beaucoup plus importants lorsque vous décrirez ce que vous voulez exactement que l'ordinateur exécute automatiquement.

QUATRIEME PARTIE

La quatrième partie traite de la puissance, de la vitesse et de la facilité. Les possibilités de l'ordinateur peuvent être considérablement accrues par l'apprentissage de quelques nouvelles astuces. Elles ne sont pas difficiles à apprendre et vous pouvez les aménager pour répondre à vos besoins particuliers de stockage et de sortie de documents imprimés.

Nous utiliserons des exemples désormais familiers de base de données pour définir les options de menu, utiliser les clauses « do while » et automatiser des traitements pour personnaliser votre système, afin de parvenir à gagner du temps et de l'énergie. La personnalisation de votre système vous permettra de produire des documents imprimés particuliers.


DATABASE

CHAPITRE XI

L'ART DES PROCEDURES

La plupart des systèmes de gestion de base de données ont un processeur de langage d'interrogation et un éditeur de document imprimé. Ces deux éléments du SGBD peuvent satisfaire la plupart, sinon tous, vos besoins. Tout besoin particulier s'obtient facilement en utilisant des procédures simples. Dans les chapitres précédents, nous avons étudié plusieurs exemples de procédures qui généraient des traitements spéciaux. Vous pouvez faire beaucoup de choses au moyen des procédures permettant d'effectuer automatiquement des tâches à votre place, à tout moment. Ce procédé ne fait pas seulement des sorties de données, fignolées, sur mesure, mais c'est également un moyen très pratique pour optimiser votre travail.

Pour illustrer l'économie de travail que peut réaliser une procédure, nous travaillerons avec un exemple simple d'une base de données d'un registre de chèques. Le plan de cet exemple de base de données B:REGCHEQ est montré à la Figure 11-1.

CHAMP	DESCRIPTION DU CHAMP	NOM DE RUBRIQUE	TYPE	TAILLE	DECIMALE(S)
1	Numéro de chèques	CHEQNO	N	4	
2	A l'ordre de	ORDRE	C	20	
3	Montant du chèque ou dépôt	MONTANT	N	7	2
4	Dépôt ou chèque	DEPOT	L	1	
5	Déductible (O/N)	DEDUCT	L	1	
6	Débité (O/N)	DEBIT	L	1	
7	Date (JJ/MM/AA)	DATE	C	8	

Figure 11-1. Plan d'une base de données registre de chèques

Dans cet exemple particulier, nous avons choisi d'avoir une rubrique logique indiquant si le montant est un chèque ou un dépôt. Un « Y » indique que le montant est un dépôt.

Pour calculer la balance du compte, nous procéderons au moyen d'un dialogue en langage d'interrogation comme à l'Ecran 11-1.

```
.SUM MONTANT TO MDEPOT FOR DEPOT
10677.80
.SUM MONTANT TO CHEQ FOR.NOT.DEPOT
9450.51
.STORE MDEPOT-CHEQ TO BALANCE
1227.29
```

Ecran 11-1

Notez que la variable qui stocke la somme des dépôts est appelée MDEPOT. Ce serait une erreur que d'appeler cette variable DEPOT. Une rubrique de données et une variable mémoire ne peuvent avoir le même nom. L'ordinateur ne s'y retrouverait pas.

Notez : Dans cet exemple, nous avons ajouté tous les chèques et tous les dépôts chaque fois que nous calculons la balance. Dans un livre de comptes conventionnel, nous conservons habituellement une balance courante. Avec l'ordinateur, il est généralement plus facile de recalculer la balance à chaque fois plutôt que de conserver une balance courante.

Comme vous pouvez le voir, il suffit de trois instructions pour déterminer la balance de la banque. Nous aimerions connaître fréquemment la balance de la banque. Pour économiser du travail et l'effort de frappe de trois instructions, chaque fois que nous voulons la balance, nous ferons en sorte que l'ordinateur se rappelle les instructions. Une « procédure » est le moyen qui permet à l'ordinateur de « se souvenir ».

L'ordinateur « se souvient » d'une procédure, comme dans notre exemple, si l'on place les instructions dans un fichier spécial. Ce fichier est placé sur le disque et est disponible lorsque nous voulons l'utiliser. Avec dBASE II, ce fichier spécial s'appelle un FICHIER DE COMMANDE. Un fichier de commande fournit la possibilité de « sauvegarder » un groupe de commandes — comme un ensemble — afin d'éviter de les retaper à chaque fois.

Pour permettre à l'ordinateur de se rappeler une procédure, nous devons d'abord lui indiquer que nous voulons l'écrire. Avec dBASE II, on y parvient avec la commande MODIFY COMMAND.

```
.MODIFY COMMAND
```

L'ordinateur répond par la demande du nom de fichier. Les règles de nom de fichier s'appliquent comme pour les fichiers de base de données et les fichiers d'état déjà vus précédemment. Un nom de fichier doit avoir de une à huit lettres et doit commencer par une lettre. Vous devez également identifier chaque lecteur de disque sur lequel est stockée la procédure en utilisant l'identificateur de lecteur de disque. Dans cet exemple, nous appellerons ce fichier de commande BALANCE et le placerons dans le lecteur B.

```
ENTREZ LE NOM DE FICHIER : B:BALANCE
```

L'écran s'efface. Pendant un très court moment, l'écran affiche :

```
NOUVEAU FICHIER (NEW FILE)
```

Les mots « NOUVEAU FICHIER » disparaîtront et l'écran deviendra complètement vierge à l'exception du curseur dans le coin supérieur gauche de l'écran. Il n'y a pas de point en début de ligne. Ce n'est pas un problème — c'est simplement la manière dont le concepteur a implémenté cette commande.

Tapez alors les instructions, les unes après les autres, comme si vous tapiez avec une machine à écrire sur une feuille de papier. Les instructions apparaissent sur l'Ecran 11-2.

Notez qu'une ligne supplémentaire a été ajoutée à la fin de la procédure. Cette ligne contient le simple mot CANCEL (ANNULER) qui redonne le contrôle de l'ordinateur au clavier après la fin d'exécution de la procédure.

```
SUM MONTANT TO MDEPOT FOR DEPOT
SUM MONTANT TO CHEQ FOR.NOT.DEPOT
STORE MDEPOT-CHEQ TO BALANCE
CANCEL
```

Ecran 11-2

L'ordinateur n'exécutera pas les instructions pendant que vous écrirez la procédure. Comme nous ne sommes pas de parfaites dactylos, le système fournit le moyen de modifier le texte par l'utilisation des touches de contrôle. Les possibilités de l'éditeur et les touches de contrôle associées sont présentées à la Figure 11-2.

CONTROLE	ACTION D'EDITION
Ctrl-S ou ←	Déplace le curseur d'un caractère vers la gauche
Ctrl-D ou →	Déplace le curseur d'un caractère vers la droite
Ctrl-E ou ↑	Déplace le curseur d'une ligne vers le haut
Ctrl-X ou ↓	Déplace le curseur d'une ligne vers le bas
Ctrl-N	Crée une nouvelle ligne
Ctrl-T	Supprime nouvelle ligne
Ctrl-Y	Efface le contenu d'une ligne
Ctrl-G CLR ou DEL →	Efface un caractère
Ctrl-V	Insère des caractères
Ctrl-W	Sauvegarde la procédure sur disque et revient au clavier pour la suite des opérations
Ctrl-Q →	Abandonne la procédure et revient au clavier pour la suite des opérations·

Figure 11-2. Possibilités d'édition de MODIFY COMMAND

Lorsque les commandes viennent d'être tapées au clavier (Ecran 11-2), tapez simplement CONTROL et W en même temps. Cela aura pour effet d'écrire sur le lecteur B le fichier de commande appelé « BALANCE » : vous pouvez l'utiliser aussi souvent que vous le souhaitez.

Vous pouvez mettre en oeuvre la procédure B :BALANCE chaque fois que la base de données registre des chèques est sélectionnée. Pour que l'ordinateur exécute la procédure B :BALANCE, tapez « DO » suivi de B :BALANCE, comme sur l'Ecran 11-3. Vous voyez que seul le résultat est affiché, et non pas les instructions elles-mêmes.

C'est certainement plus facile que de retaper les instructions à chaque fois que vous voulez obtenir la balance.

```
.DO B:BALANCE
10677.80
9450.51
1227.29
*** Abandon du traitement en cours ***
◆
```

Ecran 11-3

C'est fantastique. Nous avons seulement tapé quelques mots (DO :B :BALANCE) et la réponse apparaît. Nous avons tout de même un petit problème : à quoi correspond chaque nombre présenté à l'écran ? La solution est simple. Nous pouvons avoir l'affichage des instructions en même temps que les réponses (Ecran 11-1). Les instructions sont affichées en utilisant la commande « SET ECHO ON ». En général, on n'affiche pas les commandes. Cependant, dans certains cas, il y a des situations où il est souhaitable de les voir.

Comme nous voulons ajouter à notre « procédure » la demande de visualisation de la séquence d'instructions, nous insérons « SET ECHO ON » dans notre procédure existante B :BALANCE au moyen de la commande « MODIFY COMMAND ».

```
MODIFY COMMAND
 ENTRER LE NOM DE FICHIER : B:BALANCE
```

L'écran s'efface et le fichier existant B :BALANCE sera affiché. Le curseur sera positionné dans le coin supérieur gauche de l'écran, directement sur le « S » du premier SUM. La commande SET ECHO ON est placée sur la première ligne en appuyant sur CTRL N — qui fournit une ligne vierge — et en tapant ensuite SET ECHO ON. L'utilisation de CTRL X déplacera le curseur sur le premier « C » du mot CANCEL. Appuyez sur CTRL N, ce qui insérera une nouvelle ligne, et tapez ensuite SET ECHO OFF (la condition normale). Nous obtenons alors l'écran 11.4.

```
SET ECHO ON
SUM MONTANT TO MDEPOT FOR DEPOT
SUM MONTANT TO CHEQ FOR.NOT.DEPOT
STORE MDEPOT-CHEQ TO BALANCE
SET ECHO OFF
CANCEL
◆
```

Ecran 11-4

Appuyez sur CTRL W pour sauvegarder le nouveau fichier de commande BALANCE sur le lecteur du disque B. Maintenant, si vous exécutez la procédure B :BALANCE, les résultats apparaîtront comme à l'Ecran 11-5.

```
.SUM MONTANT TO MDEPOT FOR DEPOT
10677.80
.SUM MONTANT TO CHEQ FOR.NOT.DEPOT
9450.51
.STORE MDEPOT-CHEQ TO BALANCE
1227.29
SET ECHO OFF
```

Ecran 11-5

L'écriture d'une procédure utilise les mêmes « mécanismes » que l'écriture d'une lettre. La différence, bien entendu, c'est qu'une procédure est écrite dans un langage compréhensible par l'ordinateur. La plupart des systèmes de gestion de base de données micro-informatiques vous permettent d'écrire des procédures de deux manières : soit en utilisant un éditeur propre au SGBD soit en utilisant un système de traitement de texte séparé du SGBD.

Un système de traitement de texte est un logiciel spécial conçu spécialement pour travailler avec du texte.

Lorsque vous utilisez un système de traitement de texte pour écrire une procédure, vous devez préciser un IDENTIFICATEUR DE FICHIER en plus du nom de fichier et l'identificateur du lecteur de disque. Un IDENTIFICATEUR DE FICHIER consiste en un point suivi de trois lettres. Lorsque nous travaillons « à l'intérieur » de dBASE II, l'identificateur de fichier est déclaré automatiquement — dBASE « sait » quel type de fichier vous êtes en train de créer à partir des commandes que vous lui fournissez et ajoute l'identificateur .CMD à votre place. Lorsque nous travaillons avec un système de traitement de texte, l'identificateur de fichier doit être indiqué comme une partie du nom. Le système de traitement de texte ne sait pas de quel type de fichier il s'agit.

Avec dBASE II, l'identificateur de fichier pour les procédures (les fichiers de commande) est « .CMD ». Cet IDENTIFICATEUR DE FICHIER permet au système de base de données de savoir que ce fichier est une procédure. Notre petite procédure (fichier de commande) pourrait s'appeler :

B :BALANCE.CMD

si elle était préparée sur un système de traitement de texte.

Même si la procédure peut être rédigée à partir du système de base de données, il est souvent avantageux de l'écrire avec un système de traitement de texte. Les systèmes de traitement de texte offrent généralement des possibilités d'édition plus puissantes que celles disponibles avec le SGBD. Sinon à quoi serviraient-ils. Ces fonctions particulières d'édition sont très utiles lorsque l'on écrit de longues procédures.

Les systèmes de traitement de texte fournissent habituellement un mode spécial pour l'écriture de commandes pouvant être lues par l'ordinateur. Toutes les procédures, les fichiers de commande et les programmes d'ordinateur préparés sur un système de traitement de texte, doivent être écrits en utilisant ce mode. Le mode normal

du système de traitement de texte sert à rédiger des documents et des lettres, et à ajouter au texte des symboles invisibles. Ces symboles servent à l'usage interne du programme et, par voie de conséquence, pour imprimer ces lettres et ces documents. Ces symboles supplémentaires peuvent poser des problèmes quand ils sont inclus par inadvertance dans une procédure. Si vous n'êtes pas sûr du mode à utiliser, consultez le manuel d'utilisation de votre système de traitement de texte.

Economisez du temps et vos nerfs

Lorsque l'ordinateur exécute une instruction telle que SUM, il doit « lire » l'ensemble de la base de données à partir du disque. Ce temps de lecture de la base de données dépend principalement de la taille de la base et du type de lecteur de disque. Un lecteur de disque souple classique peut lire la base de données à 1 600 caractères par seconde. La lecture d'une petite base de données de 16 000 caractères prendra donc 10 secondes. Une base de 160 000 caractères prendra 100 secondes.

Si vous entrez les commandes manuellement, vous remarquerez qu'un court délai de lecture de la base devient très ennuyeux si le délai dépasse 5 secondes. Il est par conséquent plus avantageux de grouper les commandes pour avoir un meilleur temps de « lecture ». Si vous avez à entrer un certain nombre de commandes manuelles, il est souvent préférable de les placer dans une procédure. Cela vous libérera des courts temps d'attente (d'une à plusieurs secondes) entre chaque instruction.

L'idée sous-tendue par l'établissement d'une procédure est de faire faire tout le travail (ou au moins l'essentiel) par l'ordinateur. Vous lui donnez une liste d'instructions élémentaires — il vous les exécute sur le champ à votre place. Vous êtes le maître, il est l'esclave fidèle et complaisant. Si vous vous asseyez au clavier de l'ordinateur et l'interrogez sur le contenu de votre base de données, vous pouvez gagner un temps fou en écrivant une procédure pour que l'ordinateur exécute à votre place.

Par exemple, supposons que nous voulions acquérir un grand nombre d'informations sur le scotch et le gin de la base de données stock de notre boutique d'alcools. Nous pouvons nous asseoir au clavier et taper une série de commande OU, nous pouvons écrire une procédure qui sera composée de la liste de ces commandes et l'ordinateur fera le travail à votre place.

```
COUNT FOR ALCOOL='SCOTCH'
COUNT FOR ALCOOL='GIN'
COUNT FOR ALCOOL='SCOTCH'.AND.CONT=50 CL'
etc.
```

Si vous entrez ces commandes à partir du clavier, l'ordinateur prendra quelques secondes pour répondre à chacune d'entre elles. Si la base de données est très importante, cela peut prendre plusieurs secondes pour chaque réponse. Si vous entrez les commandes sous forme d'une procédure, vous pouvez vous détendre et prendre une tasse de café pendant que l'ordinateur vous fournit les réponses que vous attendez. L'ordinateur peut prendre moins de temps mais VOUS n'êtes pas assis en train de pianoter pendant que l'ordinateur cherche les réponses.

Nous sommes tous familiers avec une suite d'instructions devant être suivies explicitement. L'ordinateur utilise ce type d'instructions. Pour que l'ordinateur soit capable de suivre la procédure, nous devons écrire les instructions de manière compréhensible pour l'ordinateur. Rappelez-vous que l'ordinateur peut être très, très rapide, mais il n'est pas très clairvoyant. Une procédure d'ordinateur doit être écrite de façon claire. Nous ne devons jamais supposer que l'ordinateur « sait ».

Pour bien comprendre cela, voyons l'exemple suivant. Si nous demandons à un très jeune enfant de compter jusqu'à trois, il le fera certainement. Apprendre à l'ordinateur à compter jusqu'à trois est quelque chose de différent. L'ordinateur peut seulement compter jusqu'à trois si nous lui apprenons — nous lui fournissons une procédure — pour compter jusqu'à trois. Même si on le fait une première fois, il ne sera pas capable de le refaire à moins que l'on réutilise cette procédure.

Examinons une version en langage courant d'une procédure d'ordinateur pour compter jusqu'à trois (ci-dessous). Cette procédure ressemble à un « pense-bête » qu'une personne pourrait suivre avec une calculette à mémoire.

Etape 1. Ranger « zéro » en mémoire
Etape 2. Ajouter 1 au contenu de la mémoire et ranger le résultat en mémoire
Etape 3. Afficher le contenu de la mémoire
Etape 4. Ajouter 1 au contenu de la mémoire et ranger le résultat en mémoire
Etape 5. Afficher le contenu de la mémoire
Etape 6. Ajouter 1 au contenu de la mémoire et ranger le résultat en mémoire
Etape 7. Afficher le contenu de la mémoire

Lorsque nous voulons faire exécuter cette procédure par l'ordinateur, nous devons l'écrire de façon différente. La « conversion » de notre langage usuel en un langage

informatique sera différente pour chaque langage d'ordinateur, comme il différerait si nous traduisions ce texte dans une langue comme l'Anglais, l'Allemand, l'Espagnol. Avec le langage de développement d'application de dBASE II (ADL), on obtient :

Etape 1. STORE 0 TO X
Etapes 2 et 3. STORE X+1 TO X
Etapes 4 et 5. STORE X+1 TO X
Etapes 6 et 7. STORE X+1 TO X

Cette procédure contient uniquement des termes commençant par le mot STORE. Les numéros d'étapes sont présentés de manière à voir facilement la correspondance entre les deux versions de la procédure.

Vous pouvez noter deux choses au sujet de cette procédure :

- Premièrement, nous faisons la même chose plusieurs fois de suite.
- Deuxièmement, cette solution ne conviendrait pas si nous avions à faire la même chose un grand nombre de fois — par exemple compter jusqu'à mille.

La version en langage usuel peut être ré-écrite comme suit :

Etape 1. Ranger « zéro » en mémoire
Etape 2. Ajouter 1 au contenu de la mémoire et ranger le résultat en mémoire
Etape 3. Afficher le contenu de la mémoire
Etape 4. Répéter l'Etape 2
Etape 5. Répéter l'Etape 3
Etape 6. Répéter l'Etape 2
Etape 7. Répéter l'Etape 3

Dans un livre de A. Milne, « Winnie The Poo », Winnie et Piglet découvrent des traces de pas dans la neige devant la maison de Piglet. Rêvant de grande aventure, les deux personnages se mettent à suivre les traces pour voir où elles pourraient les conduire et quel type de créature a bien pu les laisser.

Les traces de pas qu'ils suivent finissent par les conduire derrière la maison de Piglet. Là, ils découvrent trois jeux d'empreintes de pas qui partent de la maison de Piglet. L'une des empreintes est plus petite que les deux autres. Après une discussion, ils imaginent qu'ils suivent un woozle et un wizzle. Puis ils suivent de nouveau les traces de pas. Après un certain temps, ils trouvent encore deux types d'empreintes de pas qui ont rejoint les trois autres. Piglet commence vraiment à s'interroger et découvre qu'ils sont en train de faire le chemin qui conduit à la maison. Ils le quittent. Winnie the Poo s'aperçoit finalement que les empreintes de pas sont les leurs... Ils avaient marché de façon circulaire.

Le cas de Winnie the Poo et Piglet, tournant en rond, aurait pu devenir une grande aventure, mais ne conduisait nulle part. Dans notre cas, cependant, il est clair que tourner en rond va nous permettre d'aller plus vite. Les cercles facilitent l'écriture de procédures simples.

DO WHILE... ENDDO (faire tant que... fin faire)

Simplification 1

Etape 1. Ranger « zéro » en mémoire
Etape 2. Ajouter 1 au contenu de la mémoire et ranger le résultat en mémoire
Etape 3. Afficher le contenu de la mémoire
Etape 4. Si la mémoire est inférieure à 3, aller à l'Etape 2

Simplification 2

Etape 1. Ranger « zéro » en mémoire
Etape 2. Faire l'Etape 3 et 4 aussi longtemps que le contenu de la mémoire sera inférieur à 3
Etape 3. Ajouter 1 au contenu de la mémoire et ranger le résultat en mémoire
Etape 4. Afficher le contenu de la mémoire

Le premier exemple ressemble à la démarche d'écriture d'une procédure dans un des langages d'ordinateur « traditionnel » tels que le FORTRAN, le COBOL ou le BASIC.

L'exemple suivant est représentatif des langages « modernes » tels que le PASCAL, PL/1 et ADL. Si nous écrivons notre simple exemple de comptage (Simplification 2) en ADL, nous obtenons :

```
STORE 0 TO X
DO WHILE X<3
STORE X+1 TO X
ENDDO
```

Le symbole « < » est une abréviation arithmétique signifiant « inférieur à ». La ligne commençant par un DO se lit « faire tant que X est inférieur à 3 ». Le symbole (« > ») signifie « supérieur à ». La ligne peut s'écrire : « DO WHILE 3>X », qui se lit « faire tant que 3 est supérieur à X ». Les deux lignes DO WHILE X<3 et DO WHILE 3>X ont exactement la même signification.

Ce nouvel exemple a autant d'instructions que l'original. Il y a cependant une différence importante — nous pouvons faire compter l'ordinateur jusqu'à cent, mille ou jusqu'à un million simplement en changeant le « 3 » par le nombre que nous désirons atteindre dans le comptage. La commande DO WHILE X<3 indique à l'ordinateur que vous souhaitez voir répéter les instructions suivantes aussi longtemps que la valeur de X sera inférieure à 3. ENDDO marque la fin du groupe d'instructions commençant par DO WHILE. Chaque DO WHILE doit avoir un ENDDO. DO WHILE — ENDDO est le moyen pour dire à l'ordinateur d'exécuter un même jeu d'instructions aussi longtemps qu'une même condition (telle que X<3 :) est vraie. Le groupe de lignes commençant par un DO WHILE et finissant par un ENDDO est appelé une « BOUCLE ».

Initialiser la boucle

Immédiatement avant la boucle, nous avons utilisé une instruction qui range la valeur 0 dans la variable mémoire X. Nous savons que si nous comptons jusqu'à trois, nous commençons à un. L'ordinateur ne sait pas quelle est la valeur de départ. On doit lui indiquer où commencer — et comment compter. Cette simple instruction — STORE 0 TO X — fait deux choses :

- Elle range la valeur de 0 dans X
- Elle « crée » également la variable mémoire X

Il n'est pas permis d'utiliser une variable mémoire dans une procédure tant que cette variable n'a pas été « créée ». De même, il ne nous est pas permis de « créer » la variable mémoire sans lui fournir une valeur initiale. Dans ce cas, la variable est créée et on lui affecte une valeur initiale par l'instruction « STORE 0 TO X ». La ligne « initialise » la boucle en fournissant la position de départ et en créant la variable X.

L'accumulateur : un concept de base

Cette petite procédure de comptage est un ACCUMULATEUR. Celui-ci constitue la base des machines à additionner ordinaires et des calculatrices électroniques. C'est une notion élémentaire assez souvent utilisée dans les procédures d'applications professionnelles des systèmes de base de données.

Les procédures permettant d'accomplir des tâches plus complexes sont généralement faites d'un groupe de procédures simples. Comme exemple, supposons que nous

voulions compter de 1 à 10 et ensuite de 10 à 100. La solution sera d'utiliser deux de nos boucles de comptage, l'une à la suite de l'autre.

```
STORE 0 TO X

DO WHILE X<10
STORE X+1 TO X
ENDDO

DO WHILE X<100
STORE X+10 TO X
ENDDO
```

Une procédure alternative : « IF »

Une autre façon d'obtenir exactement le même résultat est de permettre à l'ordinateur de faire des actions différentes à partir de différentes valeurs de X. C'est, bien entendu, ce qui apparaît ci-après, cependant cette procédure a l'avantage de partir du fait que nous savons tout de X et de ce que nous voulons obtenir. L'ordinateur est capable de prendre des décisions — bien que limitées. Nous pouvons tirer avantage de cette possibilité et écrire une procédure équivalente.

```
STORE 0 TO X

DO WHILE X<100

 IF X<10
    STORE X+1 TO X
 ENDIF

 IF X>=10
    STORE X+10 TO X
 ENDIF

ENDDO
```

Lorsque nous voulons que l'ordinateur prenne une décision pour faire ou non quelque chose, nous utilisons le mot « IF ». On l'utilise exactement de la même façon que lorsque nous utilisons le mot SI dans la conversation courante. S'il pleut, je prends un parapluie ; si je n'ai plus d'essence, je m'arrête et je prends de l'essence. Nous utilisons SI lorsque l'action qui doit être menée (ou la conclusion qui doit être tirée) dépend d'une quelconque condition.

L'action de notre exemple est d'ajouter un nombre à un autre. La condition sera la valeur un plus l'accumulateur X. Lorsque l'ordinateur prend chaque décision, il ne sait rien sur l'autre IF. Chaque IF doit avoir un ENDIF de la même façon que chaque DO WHILE doit avoir un ENDDO. L'information qui suit IF (X<10) est la condition à partir de laquelle l'ordinateur peut prendre une décision. La décision qu'il prendra dépend si oui ou non X est inférieur à 10. Si X est inférieur, le IF s'applique et l'ordinateur exécute l'instruction STORE X+1 TO X. Si ce n'est pas le cas, l'ordinateur ne stocke pas X+1 TO X.

DO WHILE et IF sont les outils essentiels de vos différentes procédures. Ces exemples utilisent la terminologie particulière du langage de développement des applications de dBASE II. Les concepts, cependant, sont universels et sont employés dans tous les langages d'ordinateur. La terminologie ressemble à celle des langages modernes tels que PL/1 et PASCAL.

Pour vous donner une meilleure idée de l'utilisation de DO WHILE et de IF, nous allons écrire une procédure faisant la même chose que B:BALANCE en utilisant ces deux options. Nous en profiterons pour montrer comment mieux contrôler les affichages produits par l'ordinateur. Le nouveau programme B:BALANCE est présenté par l'Ecran 11-6. Les résultats originaux de la procédure (Ecran 11-1) sont reproduits ici à titre de comparaison.

```
.SUM MONTANT TO MDEPOT FOR DEPOT
10677.80
.SUM MONTANT TO CHEQ FOR.NOT.DEPOT
9450.51
.STORE MDEPOT-CHEQ TO BALANCE
1227.29
```

Ecran 11-1

```
USE B:REGCHEQ
SET TALK OFF
STORE 0 TO MDEPOT,CHEQ
DO WHILE.NOT.EOF
 IF DEPOT
     STORE MONTANT+MDEPOT TO MDEPOT
 ENDIF
 IF.NOT.DEPOT
     STORE MONTANT+CHEQ TO CHEQ
 ENDIF
SKIP
ENDDO
?'TOTAL DES DEPOTS ',MDEPOT
?'TOTAL DES CHEQUES',CHEQ
?'BALANCE          ',MDEPOT-CHEQ
SET TALK ON
CANCEL
```

Ecran 11-6

Dans notre exemple, nous avons utilisé des instructions qui peuvent vous être inconnues

```
SET TALK OFF/ON
SKIP
DO WHILE.NOT.EOF
?
```

SET TALK OFF/ON

Nous avons déjà observé que beaucoup de commandes — comme STORE, SUM, etc. — affichent une réponse chaque fois qu'elles sont utilisées. L'ordinateur vous

« parle » de cette manière. C'est souhaitable lorsque vous travaillez avec l'ordinateur à partir du clavier. Par contre, ces messages sont indésirables lorsque nous utilisons des procédures. Ils désorganisent l'écran (et/ou l'imprimante). Avec dBASE II, la réponse visuelle d'une commande peut être en fonction ou hors fonction. SET TALK ON et SET TALK OFF sont justement les commandes qui permettent la mise en ou hors fonction.

```
SKIP
```

Le système de gestion de base de données travaille actuellement avec un seul enregistrement à la fois. Lorsque la ligne USE est créée, le SGBD se positionne à l'extrémité supérieure de la base de données — Enregistrement 1. Chaque utilisation du verbe SKIP avancera le SGBD d'un enregistrement. Lorsque le dernier enregistrement est atteint, l'utilisation de SKIP permet d'indiquer au SGBD que nous venons d'atteindre la fin du fichier.

```
DO WHILE.NOT.EOF
```

Comme nous l'avons vu dans les exemples précédents, la commande DO WHILE s'applique tant qu'une quelconque condition est vraie. EOF se lit « Fin de fichier ». La commande signifie littéralement « ordinateur — exécute ce qui suit jusqu'à ce que tu rencontres la fin de la base de données ». C'est probablement l'aspect de DO WHILE le plus couramment utilisé. Cela a pour effet de stopper automatiquement la BOUCLE DO lorsque l'on atteint la fin de la base de données.

```
?
```

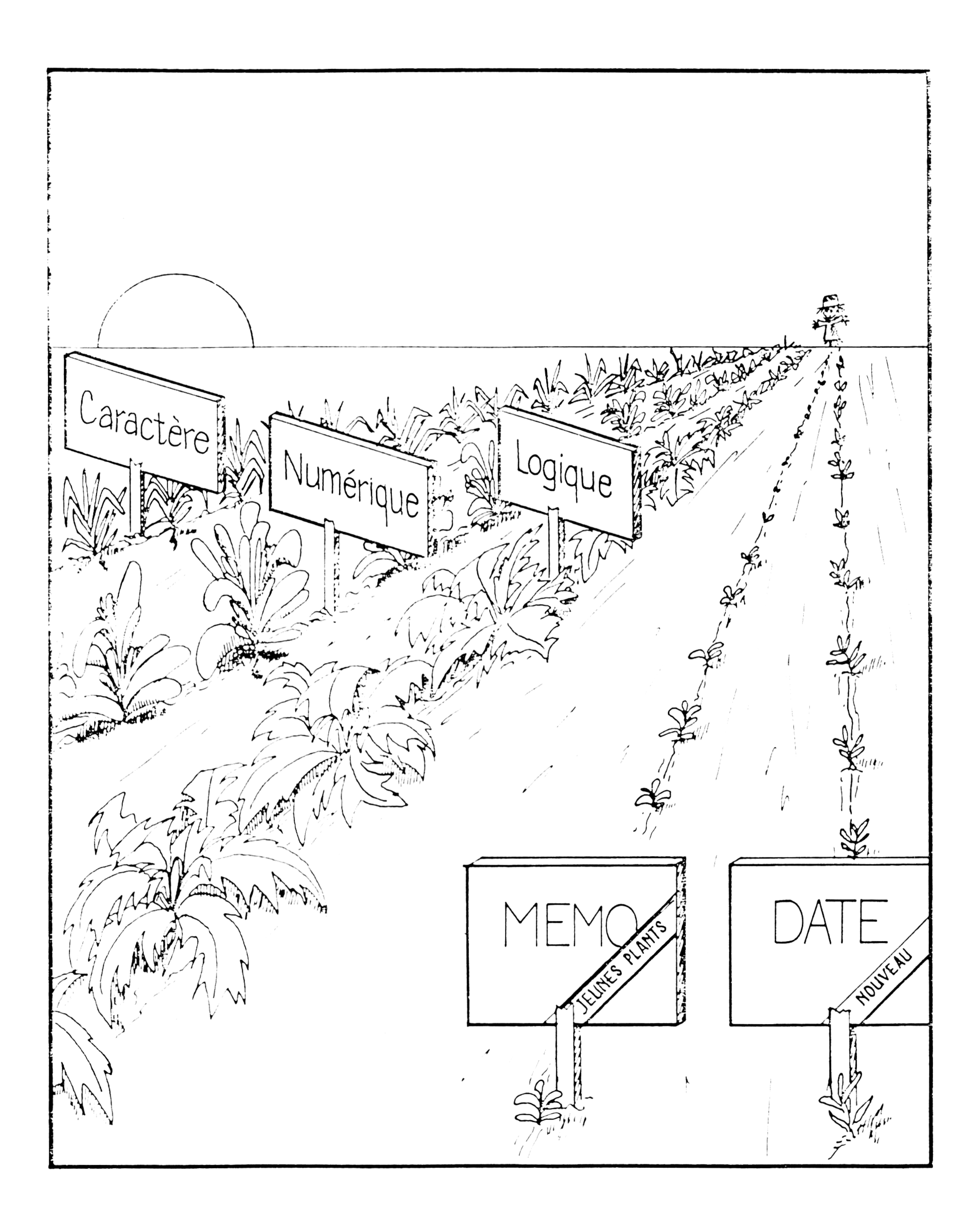
Le point d'interrogation est une commande très souple donnant la possibilité d'afficher des informations spécifiques. Les apostrophes à chaque extrémité du texte — comme dans « TOTAL DES DEPOTS » — sont appelées des délimiteurs. Leur présence indique que les caractères entre apostrophes constituent du texte destiné à l'affichage. La présence d'un nom d'une variable mémoire indique que le contenu de cette variable sera affiché. La virgule est utilisée pour séparer les éléments qui doivent être affichés. Chaque point d'interrogation produira une ligne d'affichage.

Notre nouvelle procédure B :BALANCE est un tout petit peu plus compliquée que la version originale. En un sens, quelques instants de travail supplémentaire permettent d'obtenir un résultat plus soigné. Lorsque nous commandons à l'ordinateur d'exécuter ce fichier de commande, on obtient l'affichage de l'Ecran 11-7 ci-dessous.

```
.DO B:BALANCE
TOTAL DES DEPOTS           10677.80
TOTAL DES CHEQUES           9450.51
BALANCE                     1227.29
*** Abandon du traitement en cours ***
◆
```

Ecran 11-7

Les procédures permettent à l'ordinateur de travailler beaucoup plus pour vous. C'est un moyen commode d'exploiter plus encore votre ordinateur. Dans ce chapitre, nous avons introduit un certain nombre de concepts de base : l'accumulateur, la boucle DO, la décision IF ainsi que quelques-unes des commandes dBASE II fréquemment utilisées dans les procédures. Il est assez amusant d'écrire des procédures telles que celles décrites dans ce chapitre pour finalement gagner du temps et s'économiser du travail. De plus, elles réduisent le risque d'erreur. Après avoir écrit correctement une procédure, l'ordinateur est capable d'exécuter la fonction inlassablement, sans faire d'erreur.



Caractère
Numérique
Logique
MEMO
JEUNES PLANTS
DATE
NOUVEAU

CHAPITRE XII

CREER UNE PROCEDURE QUI TRAVAILLE POUR VOUS

Les procédures sont prévues pour vous économiser des efforts, de l'argent et du temps, et vous fournir un résultat plus satisfaisant. On peut les utiliser pour la plupart des usages que vous pourriez imaginer. Les activités routinières « automatisées » sont une des choses les plus utiles que vous pouvez faire exécuter par l'ordinateur. Après tout, l'ordinateur ne doit-il pas vous rendre la vie plus facile ?

Dans ce chapitre, nous étudierons un exemple complet pour bien comprendre ce que peut faire un système de gestion de base de données et un ordinateur. Nous resterons dans le royaume de la familiarité universelle et « programmerons » un registre de chèques ordinaire. Le processus d'un registre de chèques a l'avantage de pouvoir montrer un exemple numérique.

Le plan de notre exemple de base de données B:REGCHEQ est montré à la Figure 12-1.

RUBRIQUE	DESCRIPTION DE LA RUBRIQUE	NOM DE LA RUBRIQUE	TYPE	TAILLE	DECIMALE(S)
1	Numéro du chèque	CHEQNO	N	4	
2	Payé à l'ordre de	ORDRE	C	20	
3	Montant du chèque ou dépôt	MONTANT	N	7	2
4	Dépot ou chèque	DEPOT	L	1	
5	TVA déductible (O/N)	DEDUCT	L	1	
6	Débit (O/N)	DEBIT	L	1	
7	Date (jj/mm/aa)	DATE	C	8	

Figure 12-1. Plan d'une base de données Registre des chèques

Les rubriques de ce plan correspondent à un registre de chèques ordinaire mis à part un détail important : il n'existe pas de rubrique pour la balance courante du compte. Dans un livre de chèques conventionnel, il est vital de conserver la balance courante. Il ne serait évidemment pas pratique de faire la balance de l'ensemble du livre de chèques chaque fois que nous l'utilisons. Avec l'ordinateur, cependant, ce n'est pas vrai. Il est également très raisonnable de faire la mise à jour de tous les livres chaque fois que nous utilisons le livre de chèques. C'est ce que nous avions démontré au Chapitre XI.

Il y a un certain nombre de tâches routinières induites par le tenue à jour de tout registre de chèques. Parmi celles-ci :

1. Entrer un chèque
2. Entrer un dépôt
3. Modifier une écriture pour corriger une faute
4. Examiner si nous avons enregistré un chèque particulier
5. Lister les chèques donnant lieu à une déduction de TVA
6. Déterminer la balance courante
7. Faire la balance (comparer avec les relevés de la banque).

La plupart de ces opérations sont exécutées chaque fois qu'elles sont nécessaires. Les autres sont exécutées régulièrement, soit journellement, mensuellement, ou annuellement. Elles sont toutes accomplies à partir du clavier de l'ordinateur en utilisant la base de données et le langage d'interrogation. A long terme, il est plus efficace (et vous serez plus en confiance) si la gestion de votre livre de chèques est faite en utilisant des procédures. La procédure met en place un processus approprié, vous n'avez plus à vous en occuper ensuite.

Au Chapitre XI, nous avons parlé de la procédure ainsi que des outils appropriés pour l'implémenter. Dans la suite de ce chapitre, nous utiliserons ces idées et ces outils afin d'écrire un jeu de procédures pour gérer notre livre de chèques. Ce jeu de procédures va servir à illustrer les concepts. Il ne s'agit pas toutefois d'un logiciel professionnel de gestion de tenue de comptes bancaires.

Pour atteindre notre objectif et adapter les tâches routinières du livre de chèques, nous avons besoin de sept procédures au total. Nous disposons déjà de l'une d'entre elles (numéro 6 : Déterminer la balance courante). Chacune des sept procédures aura la même taille que dans notre exemple B :BALANCE. Comme elles font toutes partie du même processus (gérer un registre de chèques), il sera raisonnable de les placer dans un menu.

Créer un menu est plus simple que notre exemple de procédure B :BALANCE. Pour avoir un aperçu de la simplicité de ce processus, nous prendrons notre liste des procédures et fabriquerons le menu directement à partir de ces éléments. La procédure de menu B :MENU est présentée par la Figure 12-2.

```
USE B:REGCHEQ
INDEX ON CHEQNO TO B:CHEQNO
USE B:REGCHEQ INDEX B:CHEQNO
SET TALK OFF
DO WHILE T
ERASE
?
?
? '                  MENU DU REGISTRE DES CHEQUES'
?
? '   1.  Entrer un chèque'
? '   2.  Entrer un dépôt'
? '   3.  Modifier une écriture pour corriger une faute'
```

```
? '  4.  Examiner si nous avons enregistré un chèque particulier'
? '  5.  Lister les chèques déductibles de TVA'
? '  6.  Déterminer la balance courante'
? '  7.  Faire la balance (comparer avec les relevés de la banque)
? '  8.  FIN DES OPERATIONS'
?
? '      APPUYEZ SUR LE CHIFFRE CORRESPONDANT A VOTRE SELECTION'
WAIT TO SELECTION
ERASE
IF SELECTION='1'
     DO B:SAISCHEQ
ENDIF
IF SELECTION='2'
     DO B:DEPOT
ENDIF
IF SELECTION='3'
     DO B:MODIF
ENDIF
IF SELECTION='4'
     DO B:CONSULT
ENDIF
IF SELECTION='5'
     DO B:DEDUCT
ENDIF
IF SELECTION='6'
     DO B:BALANCE
ENDIF
IF SELECTION='7'
     DO B:COMPBANQ
ENDIF
IF SELECTION='8'
     SET TALK ON
     CANCEL
ENDIF
ENDDO
```

Figure 12-2. Menu du registre des chèques

Encore une fois nous avons introduit un certain nombre de nouvelles instructions. Ce sont :

```
DO WHILE T
ERASE
WAIT TO SELECTION
?
DO WHILE T
```

Cela signifie littéralement « faire sans cesse ». La ligne DO WHILE signifie habituellement « faire tant que la condition qui suit est vraie ». Dans ce cas, « T » (pour « true ») est toujours vrai. Puisque nous ne souhaitons pas rester dans cette boucle infiniment, nous avons inclus l'élément 8 du menu, qui fournit un échappatoire au système de gestion de base de données.

```
ERASE
```

Cette instruction efface tout texte présent sur l'écran.

```
WAIT TO SELECTION
```

La commande WAIT stoppe momentanément la procédure. Cette procédure peut repartir en appuyant sur n'importe quelle touche. WAIT TO SELECTION signifie stocker le caractère venant d'être frappé dans la variable mémoire 'SELECTION' et fait repartir la procédure. Le contenu de SELECTION sert à indiquer à l'ordinateur ce qu'il doit faire.

```
?
```

Le point d'interrogation peut être utilisé de différentes manières. Lorsqu'on l'utilise seul, il produit une ligne vierge sur l'écran vidéo ou sur l'imprimante. On l'utilise également pour afficher le contenu de variables mémoire, de rubriques de données, ou de texte.

Les caractères encadrés par des apostrophes ou des guillemets apparaîtront comme du texte sur la ligne. Les espaces vierges peuvent être utilisés pour ajuster la position du texte sur l'écran vidéo ou sur l'imprimante.

La procédure de menu fonctionne comme suit. Le fait d'entrer la commande DO B :MENU provoque l'apparition du menu sur l'écran comme présenté à la Figure 12-1.

```
              MENU DU REGISTRE DES CHEQUES

    1.  Entrer un chèque
    2.  Entrer un dépôt
    3.  Modifier une écriture pour corriger une faute
    4.  Examiner si nous avons enregistré un chèque particulier
    5.  Lister les chèques déductibles de TVA
    6.  Déterminer la balance courante
    7.  Faire la balance (comparer avec les relevés de la banque)
    8.  FIN DES OPERATIONS


  APPUYEZ SUR LE CHIFFRE CORRESPONDANT A VOTRE SELECTION

WAITING◆
```

Ecran 12-1

Appuyez sur la touche numérique qui correspond à votre sélection. Voilà un autre cas où il n'est pas nécessaire d'utiliser la touche RETURN. Le chiffre que vous pressez est stocké dans la variable mémoire SELECTION comme un caractère. L'instruction IF qui compare votre sélection provoque l'exécution d'une procédure par l'ordinateur. Si vous choisissez par exemple l'élément 6, l'ordinateur exécutera le fichier de commande B :BALANCE.

Le premier jeu d'instructions dans la procédure de menu (Figure 12-2) indexera la base de données sur les numéros des chèques. Il ne sera pas nécessaire de ré-indexer à chaque fois que vous utiliserez le menu. L'opération d'indexation est présentée ici simplement pour montrer l'usage d'une base de données indexée.

Menu OPTION 1

En appuyant sur la touche 1, on sélectionne l'option 1 du menu, permettant la saisie de nouveaux chèques dans notre livre de chèques informatisé. On peut également obtenir la même chose par la commande APPEND, mais dans cet exemple, nous voulons illustrer la manière d'obtenir le résultat de APPEND et en même temps, fournir des messages descriptifs suffisamment clairs pour l'utilisateur par rapport à chaque élément de données.

Une procédure permettant de saisir un nouveau chèque est présentée à la Figure 12-3. Nous introduisons de nouveau quelques nouvelles commandes :

ACCEPT
INPUT
RETURN
APPEND BLANK

ACCEPT ET INPUT

ACCEPT et INPUT permettent à l'ordinateur de vous « poser des questions » pour vous inciter à entrer des données à partir du clavier pendant une procédure. INPUT est utilisé pour entrer des données numériques. ACCEPT sert à entrer des données sous forme de caractères. A part ce détail, ce sont les mêmes commandes.

La forme de la commande est illustrée par la Figure 12-3. La commande affiche le texte souhaité, crée une variable mémoire telle que MCHEQNO, et attend que vous saisissiez la donnée dans la variable.

RETURN

RETURN est similaire à CANCEL puisqu'elle termine la procédure, dans ce cas, la procédure de saisie des chèques. Cela a pour effet de redonner le contrôle au programme de menu B :MENU, en affichant le menu du registre des chèques.

CANCEL termine complètement l'opération en redonnant le contrôle de l'ordinateur au clavier.

APPEND BLANK

APPEND BLANK, souvent utilisée dans les procédures, est une variation de la commande append. APPEND est le seul moyen d'ajouter un enregistrement à la

base de données. APPEND BLANK ajoute simplement un enregistrement vierge, mais n'affiche pas l'enregistrement pour saisir les données. Dans cet exemple, un enregistrement vierge est créé, les données sont saisies dans des variables mémoire avec les commandes ACCEPT et INPUT, et ensuite les données sont transférées des variables mémoire dans l'enregistrement vierge en utilisant la commande REPLACE. Il n'est pas nécessaire d'utiliser SET TALK OFF puisque cette commande a déjà été accomplie dans la procédure B:MENU (Figure 12-2).

Dans cet exemple particulier, les messages sont affichés et les données sont saisies instruction par instruction. A partir de ce moment, la procédure n'est pas aussi efficace que APPEND puisque le curseur ne peut pas être déplacé en arrière pour corriger un élément de données précédent.

```
APPEND BLANK
INPUT 'ENTREZ LE NUMERO DU CHEQUE' TO MCHEQNO
ACCEPT 'PAYEZ A L'ORDRE DE' TO MORDRE
INPUT 'ENTREZ LE MONTANT DU CHEQUE' TO MMONTANT
ACCEPT 'ENTREZ LA DATE (jj/mm/aa)' TO MDATE
ACCEPT 'CE CHEQUE EST-IL DEDUCTIBLE (Y/N)' TO MDEDUCT
REPLACE CHEQNO WITH MCHEQNO, ORDRE WITH MORDRE, MONTANT
WITH MMONTANT, DATE WITH MDATE, DEDUCT WITH MDEDUCT, DEPOT
WITH N
RETURN
```

Figure 12-3. Une procédure préliminaire de saisie d'un chèque

Un affichage informatique similaire à APPEND (avec des messages beaucoup plus descriptifs remplaçant les noms de rubriques) serait de loin beaucoup plus souhaitable. De plus, pour une saisie plus confortable, le curseur devrait pouvoir se déplacer sur une ou plusieurs rubriques précédentes pour corriger les erreurs. On peut obtenir ce résultat, avec dBASE II, à l'aide de commandes spéciales destinées exactement à cet usage.

Le format général de la commande dBASE II est :

```
@ LIGNE,COLONNE SAY 'QUE VOULEZ-VOUS' GET NOM DE RUBRIQUE
  READ
```

Tout d'abord, la commande permet le contrôle du positionnement des données sur l'écran. La plupart des terminaux d'ordinateur ont un écran vidéo de 24 lignes de 80 caractères chacune. Les lignes sont numérotées depuis le haut, de 0 à 23. Les colonnes (positions des caractères) sont numérotées de gauche à droite, de 0 à 79. Pour afficher CECI EST UN EXEMPLE, à la ligne 5, commençant à la colonne 10, la commande s'écrira.

```
@ 5,10 SAY 'CECI EST UN EXEMPLE'
```

Le texte, « CECI EST UN EXEMPLE », sera affiché en intensité moyenne comme les noms de rubriques dans APPEND.

On peut aussi afficher le contenu d'une rubrique par :

```
@ 5,10 GET ORDRE
```

Cet exemple affiche le contenu courant de la rubrique ORDRE en double brillance, exactement comme dans la commande EDIT.

Les deux exemples peuvent être combinés :

```
@ 5,10 SAY 'CECI EST UN EXEMPLE' GET ORDRE
```

La commande READ vous permet de modifier le contenu d'une rubrique identifiée par un GET. Un exemple de procédure de saisie d'un chèque utilisant ces commandes est présenté à la Figure 12-4. L'affichage produit par cette commande est celui de la Figure 12-2.

```
APPEND BLANK
@ 5,10 SAY 'ENTREZ LE NUMERO DU CHEQU  GET CHEQNO
@ 7,10 SAY 'PAYEZ A L'ORDRE DE' GET ORDRE
@ 9,10 SAY 'ENTREZ LE MONTANT DU CHEQUE' GET MONTANT
@ 11,10 SAY 'ENTREZ LA DATE (jj/mm/aa)' GET DATE
@ 13,10 SAY 'CE CHEQUE EST-IL DEDUCTIBLE (Y/N)' GET DEDUCT
READ
REPLACE DEPOT WITH N
RETURN
```

Figure 12-4

Cette procédure permet à l'ordinateur de se comporter d'une manière similaire à APPEND mis à part les messages descriptifs qui remplacent les noms de rubriques. Toutes les touches de CONTROLE pour déplacer le curseur se comportent comme dans la commande APPEND.

Notez que toutes les rubriques ne sont pas affichées, seules les rubriques intéressantes pour cette procédure le seront.

Sur l'Ecran 12-2, on peut voir le résultat de cette procédure. Le curseur est initialement positionné sur le premier caractère de la rubrique CHEQNO, qui est prête pour la saisie de données.

```
ENTREZ LE NUMERO DU CHEQUE :♦             :

PAYEZ A L'ORDRE DE :                                :

ENTREZ LE MONTANT DU CHEQUE :             :

ENTREZ LA DATE (jj/mm/aa) :               :

CE CHEQUE EST-IL DEDUCTIBLE (Y/N) :  :
```

Ecran 12-2

Comme les numéros de chèques sont habituellement séquentiels, vous pouvez gagner énormément de temps en permettant à l'ordinateur d'entrer les numéros de chèques à votre place. La procédure révisée pour accomplir ce résultat est présentée par la Figure 12-5. Les modifications sont représentées en caractères gras.

```
GO BOTTOM
STORE CHEQNO+1 TO CNO
APPEND BLANK
REPLACE CHEQNO WITH CNO, DEPOT WITH N
@5,10 SAY '      NUMERO DU CHEQUE' GET CHEQNO
CLEAR GETS
@7,10 SAY 'PAYEZ A L'ORDRE DE' GET ORDRE
@9,10 SAY 'ENTREZ LE MONTANT DU CHEQUE' GET MONTANT
@11,10 SAY 'ENTREZ LA DATE (jj/mm/aa)' GET DATE
@13,10 SAY 'CE CHEQUE EST-IL DEDUCTIBLE (Y/N)' GET DEDUCT
READ
RETURN
```

Figure 12-5. Procédure d'entrée d'un chèque B:SAISCHEQ

La commande GO BOTTOM positionne le SGBD sur le dernier enregistrement de la base de données. Cet enregistrement contient le dernier numéro de chèque utilisé. Le numéro de chèque que vous voulez saisir doit être le nombre qui lui succède. Ce nouveau numéro de chèque est stocké temporairement dans la variable mémoire CNO par STORE CHEQNO+1 TO CNO. Un enregistrement vierge est ajouté à la base de données. Le nouveau numéro de chèque est écrit sur l'enregistrement vierge par REPLACE CHEQNO WITH CNO.

L'instruction CLEAR GETS annule les mouvements de curseur possibles vers les rubriques affichées par les commandes GET antérieures au CLEAR GETS. L'affichage fourni par cette procédure est similaire à l'Ecran 12-2, excepté que le curseur se trouve positionné sur le caractère le plus à gauche de la rubrique ORDRE et, d'autre part, le numéro du chèque est affiché à l'écran. Voir l'Ecran 12-3.

```
ENTREZ LE NUMERO DU CHEQUE : 1250 :

PAYEZ A L'ORDRE DE : ◆                              :

ENTREZ LE MONTANT DU CHEQUE :          :

ENTREZ LA DATE (jj/mm/aa) :            :

CE CHEQUE EST-IL DEDUCTIBLE (Y/N) :  :
```

Ecran 12-3

Menu OPTION 2

La procédure 2 vous permet d'entrer les dépôts dans votre livre de banque électronique. La Figure 12-6 illustre la procédure de saisie d'un dépôt, nommé B :DEPOT.

```
APPEND BLANK
REPLACE DEPOT WITH Y, ORDRE WITH 'DEPOSIT'
@9,10 SAY 'ENTREZ LE MONTANT DU DEPOT' GET MONTANT
@11,10 SAY 'ENTREZ LA DATE (jj/mm/aa)' GET DATE
READ
RETURN
```

Figure 12-6. Procédure de saisie d'un dépôt

Cette procédure est similaire, mais un peu plus simple, que B :SAISCHEQ, la procédure d'entrée des chèques. L'affichage produit par B :DEPOT est montré sur l'Ecran 12-4.

```
ENTREZ LE MONTANT DU DEPOT :            :

ENTREZ LA DATE (jj/mm/aa) :            :
```

Ecran 12-4

Ici également, cette procédure ressemble à APPEND mis à part des détails qui ont été ajoutés dans les messages descriptifs et d'autre part toutes les rubriques qui ne sont pas présentes à l'affichage.

Menu OPTION 3

La troisième option du menu permet la « correction d'erreur » d'un enregistrement déjà saisi précédemment. Avec dBASE II, on peut obtenir ce résultat soit par la commande EDIT ou BROWSE. Comme vous pouvez donc le voir, il vous est possible de rédiger une procédure similaire à EDIT, la seule différence sera l'affichage de texte descriptif à la place des noms de rubriques.

Une procédure de ce type est présentée à la Figure 12-7.

```
ACCEPT 'APPUYEZ SUR C POUR UN CHEQUE, D POUR UN DEPOT' TO FRAPPE
IF FRAPPE='C'
    ACCEPT 'ENTREZ LE NUMERO DU CHEQUE' TO RECHERCHE
    FIND &RECHERCHE
    ERASE
    @5,10 SAY '        NUMERO DU CHEQUE' GET CHEQNO
    CLEAR GETS
    @7,10 SAY 'PAYEZ A L'ORDRE DE' GET ORDRE
    @9,10 SAY 'ENTREZ LE MONTANT DU CHEQUE' GET MONTANT
    @11,10 SAY 'ENTREZ LA DATE (jj/mm/aa)' GET DATE
    @13,10 SAY 'CE CHEQUE EST-IL DEDUCTIBLE (Y/N)' GET DEDUCT
      READ
ENDIF
IF FRAPPE='D'
    ACCEPT 'ENTREZ LA DATE DU DEPOT' TO RECHERCHE
    DISPLAY FOR DEPOT.AND.DATE=RECHERCHE
    INPUT 'ENTREZ NUMERO ENREGISTREMENT A MODIFIER' TO RECHERCHE
    GOTO RECHERCHE
    ERASE
    @9,10 SAY 'ENTREZ LE MONTANT DU DEPOT' GET MONTANT
    @11,10 SAY 'ENTREZ LA DATE (jj/mm/aa)' GET DATE
    READ
ENDIF
RETURN
```

Figure 12-7. Une procédure de modification d'un enregistrement

Notre premier problème est de retrouver l'enregistrement que nous cherchons. Si nous voulons modifier l'écriture d'un chèque, c'est vraiment très simple — nous demandons à l'ordinateur de retrouver l'enregistrement qui contient le numéro du chèque correspondant à l'écriture du chèque que nous voulons modifier. Par contre, si nous avons un dépôt, il nous faudra trouver l'enregistrement d'une autre manière, puisqu'il n'y a pas de données numéro de chèque lorsque l'on saisit un dépôt.

Dans notre exemple de base de données, il n'existe pas d'identification unique d'un enregistrement pour un dépôt (excepté, bien sûr, le numéro d'enregistrement). Ceci montre bien la nécessité de concevoir préalablement la base de données. Notre manque de réflexion vient de nous mener à une solution qui tourne en rond. Il est tentant d'utiliser la date comme critère de modification. Malheureusement, même si nous n'avons jamais plus d'un dépôt par jour, nous risquons des problèmes le jour où il s'en présentera deux.

La solution consiste à utiliser le numéro d'enregistrement comme un critère. L'affichage de tous les enregistrements d'une date particulière permet l'accès au numéro d'enregistrement approprié.

La procédure comporte deux parties, sélectionnées suivant que la modification s'applique ou non à un chèque :

1. Si c'est un chèque, nous recherchons l'enregistrement qui contient son numéro, puis nous produisons le même affichage que dans la procédure de saisie de chèque. L'enregistrement est trouvé grâce à la commande FIND et la variable mémoire RECHERCHE. FIND & RECHERCHE signifie rechercher l'enregistrement correspondant au numéro du chèque stocké dans recherche. Le & indique à l'ordinateur que RECHERCHE est une variable mémoire.

2. La seconde partie intervient lorsque l'on veut faire la modification d'un enregistrement dépôt, mais pas un enregistrement de chèque. L'ordinateur demande d'abord la date qui nous concerne, affiche l'enregistrement de dépôt pour cette date, et demande ensuite le numéro d'enregistrement pour l'enregistrement de dépôt que l'on veut modifier. GOTO RECHERCHE positionne la base de données sur l'enregistrement par le numéro d'enregistrement stocké dans RECHERCHE. L'affichage lors de la modification sera le même affichage que lorsque l'on entre des informations dans un enregistrement dépôt.

Menu OPTION 4

La quatrième option du menu nous permet de consulter le contenu du registre des chèques, pour déterminer si nous avons réellement tiré un chèque particulier. La Figure 12-8 nous montre les possibilités de présentation au moyen d'une procédure simple.

```
USE B:REGCHEQ
ACCEPT 'ENTREZ LE MOIS ET L'ANNEE (mm/aa)' TO TEMPS
DISPLAY FOR $(DATE,4,2)+$(DATE,7,2)=$(TEMPS,1,2)+$(TEMPS,4,2)
USE B:REGCHEQ INDEX B:CHEQNO
WAIT
RETURN
```

Figure 12-8. Une procédure pour examiner le registre des chèques

Cette procédure produit l'affichage de tous les enregistrements pour un mois particulier. Comme le fichier index semble partager les enregistrements dépôt et les enregistrements chèques, et nous souhaitons avoir cette liste dans le temps, nous utilisons la base de données sans son fichier index.

La procédure affiche (dans l'ordre des enregistrements) les enregistrements d'un mois et d'une année particulière. Les enregistrements sont affichés quinze par quinze (caractéristique de la commande DISPLAY). Après le quinzième enregistrement, le mot WAITING apparaît. L'utilisation de n'importe quelle touche, excepté SHIFT ou CONTROLE, a pour effet d'afficher les quinze enregistrements suivants. Dans la rubrique DATE, le premier et le second caractère indiquent le mois alors que le septième et le huitième indique l'année. Dans la variable mémoire TEMPS, le premier et le deuxième caractère indiquent le mois tandis que le quatrième et le cinquième indiquent l'année.

L'instruction DISPLAY compare le quatrième, le cinquième, le septième et le huitième caractère de la date avec le premier, le second, le quatrième et le cinquième caractère de la variable TEMPS. Lorsque ces deux jeux de caractères sont identiques, l'enregistrement est affiché.

Lorsque la procédure est terminée, le fichier indexé est de nouveau choisi pour être utilisé (USE).

Cet élément du menu a de bonnes chances de générer un second menu, par conséquent, nous ne l'inclurons pas dans ce chapitre. Le second menu pourrait donner lieu à une sélection d'affichages possibles, par exemple toutes sortes de listages (REPORTS) comme ceux générés dans l'exemple de stock d'une boutique d'alcools du Chapitre II.

Menu OPTION 5

L'élément 5 du menu (listing des chèques déductibles) nous donne une bonne occasion d'utiliser la commande REPORT FORM dans un fichier de commande. La Figure 12-8A montre la procédure de listage des chèques déductibles.

```
ACCEPT 'VOULEZ-VOUS UNE COPIE IMPRIMEE (O/N)' TO DEMANDE
IF DEMANDE='O'
   REPORT FORM B:DEDUCT TO PRINT FOR DEDUCT
ENDIF
IF DEMANDE='N'
   REPORT FORM B:DEDUCT FOR DEDUCT
ENDIF
RETURN
```

Figure 12-8A

Le document qui en résulte est présenté à la Figure 12-8B. Les documents imprimés sont un moyen commode de présenter les données.

PAGE NO.00001
01/02/86

DEPENSES DEDUCTIBLES

CHEQUE NO	ORDRE	DATE	MONTANT
208	FOURNITURES	22/06/85	98.50
227	MATERIEL INFORMATIQUE	14/08/85	4011.04
234	RUBANS	21/08/85	91.00
267	ORDINATEUR STAR	24/10/85	486.38
289	DEVELOPPEMENT LOGICIEL	27/11/85	79.50
323	MICRO-ORDINATEUR CITRON	23/01/86	1257.16
324	COMMANDES MECANIQUES	23/01/86	344.50
** TOTAL **			6368.08

Figure 12-8B

Menu OPTION 6

L'élément 6 du menu détermine la balance courante du compte. Cette procédure est presque identique à B:BALANCE développée dans le chapitre précédent. La Figure 12-9 nous montre les différences.

```
USE B:REGCHEQ
STORE 0 TO MDEPOT,CHEQ
DO WHILE.NOT.EOF
  IF DEPOT
    STORE MONTANT+MDEPOT TO MDEPOT
  ENDIF
  IF.NOT.DEPOT
    STORE MONTANT+CHEQ TO CHEQ
  ENDIF
SKIP
ENDDO
@5,15 SAY 'TOTAL DES DEPOTS' GET MDEPOT
@7,15 SAY 'TOTAL DES CHEQUES' GET CHEQ
STORE MDEPOT-ÇHEQ TO BALANCE
@9,15 SAY 'BALANCE       ' GET BALANCE
CLEAR GETS
WAIT
USE B:REGCHEQ INDEX B:CHEQNO
RETURN
```

Figure 12-9. Procédure de calcul de la balance courante du compte

Nous utilisons ici la commande dBASE II @ pour l'affichage, comme sur l'Ecran 12-5. L'autre différence avec notre programme B:BALANCE est l'utilisation de la commande RETURN à la place de la commande CANCEL.

```
TOTAL DES DEPOTS :      10677.80 :

TOTAL DES CHEQUES :      9450.51 :

BALANCE           :      1227.29 :


WAITING◆
```

Ecran 12-5

Puisque cela prend moins de temps de calculer la balance en utilisant B:REGCHEQ sans son fichier index, nous n'utiliserons pas d'index dans cette procédure. La rapidité vient d'une lecture séquentielle des enregistrements, plutôt que des sauts en arrière et en avant dictés par les index. Une base de données indexée nécessite des mouvements en avant et en arrière, la tête de lecture du disque se déplaçant également en avant et en arrière.

Menu OPTION 7

L'élément 7 du menu saisit les relevés de compte de la banque. Il vous faut un moyen pour contrôler si la banque prend soin de votre compte. Cette procédure peut faire ce travail.

La balance d'un compte-chèques est souvent appelée une réconciliation. Quelle que soit la façon dont on l'appelle, elle comprend toujours quatre parties :

1. Pointer les chèques débités
2. Pointer les dépôts crédités
3. Entrer les frais de banque
4. Prendre votre balance finale, ajouter les chèques non encore tirés et soustraire les dépôts non encore enregistrés au compte, comparer le résultat à la balance de la banque.

La procédure de calcul du solde est présentée à la Figure 12-10. Cette procédure comporte également un menu permettant de prendre en charge chacune des quatre parties et de s'en occuper jusqu'à ce que vous soyez satisfait.

```
STORE ' ' TO CHOIX
DO WHILE.NOT.CHOIX='5'
ERASE
@5,15 SAY '1. Pointage des chèques débités'
@7,15 SAY '2. Pointage des dépôts crédités'
@9,15 SAY '3. Saisie des frais de banque
@11,15 SAY '4. Comparer la balance de la banque avec votre balance
@13,15 SAY '5. Revenir au Menu principal'
@20,15 SAY 'ENTREZ VOTRE SELECTION' GET CHOIX
READ
DO CASE
CASE CHOIX='1'
   DO B:BALMENU1
CASE CHOIX='2'
   DO B:BALMENU2
CASE CHOIX='3'
   DO B:BALMENU3
CASE CHOIX='4'
   DO B:BALMENU4
ENDCASE
ENDDO
RETURN
```

Figure 12-10. Une procédure de menu pour calculer le solde

Si vous comparez ce menu avec le Menu principal du registre des chèques, vous trouverez quelques différences. Ces différences vous montrent que vous pouvez obtenir le même résultat par des moyens très différents.

DO WHILE.T. est remplacé par DO WHILE.NOT.(CHOIX='5'). Si l'on tape 5, on termine la boucle DO et on retourne au Menu principal. Si l'on choisit un caractère qui n'est pas dans la sélection, le menu ré-apparaît.

Dans cet exemple, on a introduit une nouvelle commande, « DO CASE ». DO CASE a quelque chose de commun avec IF, on l'utilise souvent lorsqu'il y a un grand nombre de choix exclusifs pour des actions. Un seul CASE donne lieu à une action même si plusieurs peuvent fonctionner. C'est le premier CASE qui satisfait une condition qui donne lieu à une action.

Dans une série de IF, il est possible d'en rencontrer plusieurs qui satisfassent une condition et donnent lieu à une exécution, même si celle-ci n'était pas souhaitée.

Cette construction particulière de procédure d'un menu est préférable à celle utilisée dans l'exemple du Menu principal. Le choix du nom de la procédure permet son identification immédiate lorsque l'on veut la rappeler.

Le pointage des chèques débités

Le premier élément du menu permet le « pointage » des chèques débités par la banque. Dans l'exemple de procédure de la Figure 12-11, l'enregistrement du chèque est retrouvé et la donnée pertinente est affichée sur l'écran comme un « chèque ». Si l'enregistrement qui s'affiche est celui que vous cherchez, vous « pointez » l'enregistrement. L'affichage représentatif de cette procédure est montré par l'Ecran 12-6.

```
STORE '  ' TO SUITE
STORE 0 TO NUMERO,TOTAL
DO WHILE.NOT.SUITE='X'
  ERASE
  STORE '    ' TO RECHERCHE
  @5,10 SAY 'ENTREZ LE NUMERO DU CHEQUE' TO RECHERCHE
  READ →@ 5,10 SAY 'ENTREZ LE NUMERO DU CHEQUE' GET RECHERCHE
  SEEK RECHERCHE FIND & RECHERCHE
  @7,10 SAY 'CHEQUE PAYE A L'ORDRE DE' GET ORDRE
  @9,10 SAY 'LE MONTANT EST :' GET MONTANT
  @7,50 SAY 'DATE' GET DATE
  CLEAR GETS
  @9,50 SAY 'DEBITE (Y/N)' GET DEBITE
  READ
  IF DEBIT
     STORE TOTAL+MONTANT TO TOTAL
     STORE NUMERO+1 TO NUMERO
     @12,10 SAY 'NOMBRE DE CHEQUES CREDITES' GET NUMERO
     @12,40 SAY 'TOTAL' GET TOTAL
  ENDIF
  CLEAR GETS
  @ 15,10 SAY 'ENTREZ X SI TERMINE-RETURN POUR CONTINUER' GET SUITE
  READ
ENDDO
RETURN
```

Figure 12-11. Une procédure pour "le pointage des chèques tirés"

```
ENTREZ LE NUMERO DU CHEQUE : 1221 :

CHEQUE PAYE A L'ORDRE DE : COMPAGNIE DE TELEPHONE : DATE : 29/11/86

LE MONTANT EST :   34.56 :                    DEBITE (Y/N) : Y :

NOMBRE DE CHEQUES DEBITES :  34 :                   TOTAL : 574.66 :

ENTREZ X SI TERMINE - RETURN POUR CONTINUER :◆
```

Ecran 12-6

Lorsque vous avez terminé de pointer les chèques débités, le nombre de chèques débités et leur total doit correspondre avec le relevé de la banque. Si ce n'est pas le cas, vous devez déterminer pourquoi il y a une différence. La procédure peut être répétée si nécessaire.

Le pointage des dépôts

Après avoir vu la procédure de pointage des chèques tirés, nous passons au pointage des dépôts. B:BALMENU2 est la procédure qui va exécuter cette tâche. Elle est présentée à la Figure 12-12.

```
STORE T TO SUITE
STORE 0 TO NR,DEP
DO WHILE SUITE
  ERASE
  REMARK LORSQUE C'EST TERMINE TAPEZ · STOP · A LA PLACE DU NUMERO
  REMARK D'ENREGISTREMENT
  DISPLAY MONTANT,DATE FOR DEPOT.AND..NOT.DEBIT
  ACCEPT 'ENTREZ LE NUMERO D'ENREGISTREMENT A POINTER ' TO F1
    IF F1='STOP'
      STORE F TO SUITE
    ELSE
      STORE VAL(F1) TO RECHERCHE
      GOTO RECHERCHE
      STORE NR+1 TO NR
      STORE DEP+MONTANT TO DEP
      REPLACE DEBIT WITH Y
    ENDIF
ENDDO
ERASE
@10,10 SAY 'NOMBRE DE DEPOTS POINTES' GET NR
@15,10 SAY 'LA VALEUR TOTALE ETAIT DE' GET DEP
WAIT
RETURN
```

Figure 12-12. Une procédure pour "pointer" les dépôts

Dans cet exemple, nous interrompons la boucle DO dans la procédure. La boucle DO opère aussi longtemps que la variable mémoire SUITE est vraie. Il est à noter que T et F (vrai et faux) n'ont pas besoin d'être encadrés par des points. L'ordinateur les prend implicitement pour des valeurs logiques. T veut dire vrai (true) et F veut dire faux (false).

Notez également que le numéro d'enregistrement a été stocké comme un élément caractère. (ACCEPT est un moyen pour indiquer à l'ordinateur que les données

saisies sont du type caractère). Si nous utilisons INPUT, nous ne pouvons pas saisir le mot « STOP » dans la variable mémoire F1. De cette manière, il n'est pas adéquat d'utiliser T ou F comme des noms de variables mémoire. L'ordinateur risque de les confondre à un moment ou à un autre. Dans ce cas, si nous utilisons F à la place de F1, la ligne STORE F TO SUITE sera ambiguë. De quel « F » s'agit-il ?, celui qui signifie faux, ou le contenu de F ?

D'autre part, comme nous avons stocké le numéro d'enregistrement comme une chaîne de caractères, nous avons besoin de le convertir en retour en un nombre pour pouvoir utiliser la commande GOTO. On utilise pour cela la commande STORE VAL(F1) TO RECHERCHE. Ce qui veut dire « stockez la valeur de F1 dans la variable mémoire RECHERCHE ». Comme vous pouvez le noter, les nombres peuvent être convertis en caractères et les nombres stockés sous forme de caractères peuvent être reconvertis en nombres.

Saisir les frais de banque

La troisième procédure de ce menu nous permet de saisir les frais de banque dans la base de données. Cette procédure est présentée à la Figure 12-13.

```
ERASE
APPEND BLANK
@5,5 SAY 'ENTREZ LE MONTANT DES FRAIS DE BANQUE' GET MONTANT
@7,5 SAY 'ENTREZ LA DATE (jj/mm/aa)' GET DATE
READ
REPLACE DEPOT WITH N, ORDRE WITH 'FRAIS DE BANQUE', DEBIT WITH Y
RETURN
```

Figure 12-13. Une procédure de saisie des frais de banque

Vous pouvez remarquer que les frais ont été traités comme des chèques de telle façon que les calculs fonctionnent correctement. Les frais et les chèques sont tous deux des débits du compte. Les frais sont également « pointés » au moment de la saisie puisque l'on place le caractère Y dans la rubrique DEBIT.

De nos jours, beaucoup de banques offrent différentes possibilités de crédits. Pour cette raison, un « vrai » menu de registre des chèques doit inclure une saisie pour les crédits de la banque.

L'élément 4 du menu permet la comparaison de notre version du montant de la balance aux conclusions de la banque. Cette procédure est présentée à la Figure 12-14.

```
ERASE
INPUT 'ENTREZ LA BALANCE DE LA BANQUE' TO BANQBAL
SUM MONTANT TO DEB FOR DEBIT.AND..NOT.DEPOT
SUM MONTANT TO CRED FOR DEBIT.AND.DEPOT
STORE CRED-DEB TO MABALANCE
@10,10 SAY 'MA VERSION DE LA BALANCE' GET MABALANCE
STORE BANQBAL-MABALANCE TO ERREUR
@15,10 SAY 'L'ERREUR DE LA BANQUE REPRESENTE' GET ERREUR
CLEAR GETS
WAIT
RETURN
```

Figure 12-14. Procédure de balance mensuelle

Dans cet exemple, nous ne prenons pas en considération les chèques n'ayant pas encore été tirés et les dépôts. Vous faites cela normalement sur un registre de chèques sur « papier » pour obtenir votre balance « courante ». Dans le livre de chèques de l'ordinateur, nous ignorons les chèques non tirés et travaillons avec le même jeu de données qu'utilise la banque. Si toutes les données sont entrées correctement, le résultat à l'affichage ressemble à celui de l'Ecran 12-7.

```
        ENTREZ LA BALANCE DE LA BANQUE :   1227.29

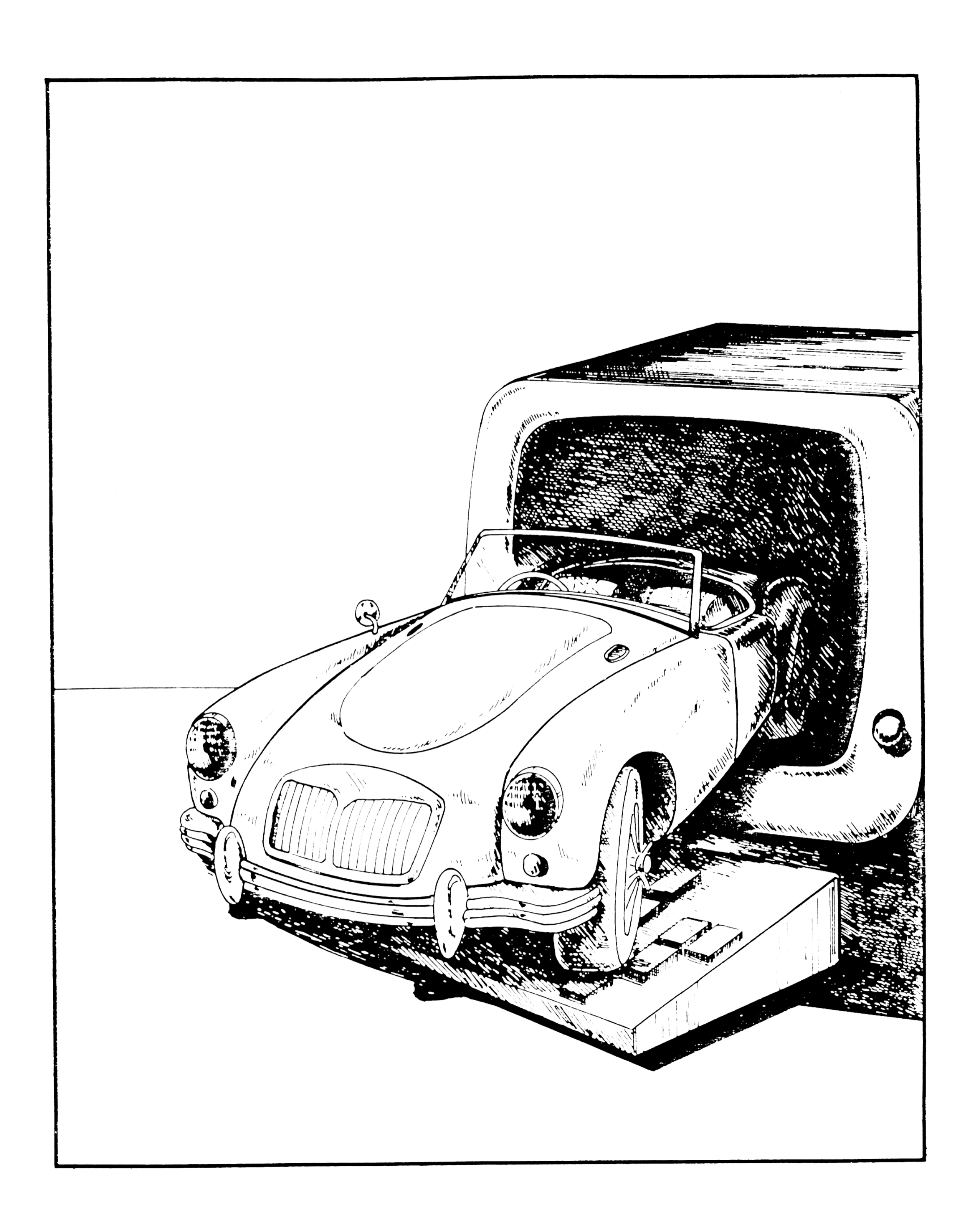
              MA VERSION DE LA BALANCE  : 1227.29 :

        L'ERREUR DE LA BANQUE REPRESENTE :    0 :

     WAITING◆
```

Ecran 12-7

Dans ce chapitre, nous avons essayé de décrire un système complet de base de données pour gérer un registre de chèques. Afin de décrire certains concepts de base, certains détails ont été négligés. Le système de menu est un moyen pratique pour faire exécuter des tâches routinières à l'ordinateur, quotidiennement, chaque semaine, ou sur d'autres périodes. Un menu peut se composer de sous-menus pour des tâches plus complexes. Comme règle, il faut retenir que les éléments de menu les plus fréquemment utilisés doivent se trouver dans la partie supérieure et ceux les moins fréquemment utilisés doivent se trouver placés parmi les derniers éléments.



CHAPITRE XIII

PROCEDURES POUR MINIMISER LES ERREURS ET LA MONOTONIE

L'efficacité de votre base de données dépend fortement de l'exactitude des données. Les données sont habituellement saisies par des personnes et celles-ci font des erreurs. Nous avons déjà brièvement parlé dans ce livre comment se servir de l'ordinateur pour accélérer et rendre moins monotone la saisie des données. Il est souvent plus tentant de se préoccuper de la « vitesse » que du côté « attractif » de la saisie. L'objectif est certainement de réduire les coûts de saisie de données. Malheureusement, toute « économie » apparente est souvent un leurre.

Les aides de l'ordinateur sont valables uniquement si elles augmentent les performances de saisie sans augmenter les erreurs. La plupart des erreurs de saisie de données sont soit typographiques, soit des erreurs d'écriture (par exemple, votre œil saute une ligne pendant le transfert des données du papier à l'ordinateur).

Les erreurs typographiques sont souvent (mais pas toujours) des erreurs graves. « Scotch » peut être saisi « scitch ». Les menus permettent quelquefois d'éviter de telles erreurs d'orthographe. Par contre, si vous appuyez sur la mauvaise touche de sélection dans un menu et si vous demandez « whiskey » au lieu de « scotch », ce sera beaucoup plus sérieux qu'une simple erreur d'orthographe. Vous pouvez mettre en place un contrôle de ce type d'erreur :

```
VOUS AVEZ CHOISI SCOTCH
     CORRECT (Y/N)
```

Maintenant, c'est très joli, mais cela prend beaucoup plus de temps que de taper « scotch » du premier coup. Vous souhaitez une saisie de données qui soit très rapide. Il est toutefois plus important de souhaiter aller doucement et correctement. Un environnement de travail plaisant permettra à la fois une saisie plus rapide et plus précise qu'un tas de gadgets.

Néanmoins, la plupart des tâches de saisie sont de vraies routines. Il est donc raisonnable de les automatiser de façon à réduire leur monotonie et de fournir une protection contre les erreurs.

La protection la plus simple contre les erreurs est la saisie d'une « donnée impossible ». Si c'est une erreur de cette sorte, la donnée qui vient d'être tapée sort des limites de valeurs permises. Par exemple, si vous entrez les données d'un élève de classe élémentaire, et auquel on ne pourra lui affecter seulement qu'un certain nombre de classes. L'ordinateur pourra donc se protéger contre la saisie d'un numéro de classe incohérent. Il ne vous évitera pas d'entrer accidentellement un numéro de classe possible (mais incorrect).

Comment se protéger alors contre les « données impossibles » ? Ce n'est pas particulièrement banal mais c'est relativement facile. Supposons que nous ayons une petite procédure pour entrer le nom, le numéro de la classe et le niveau d'un élève dans une base de données. Pour simplifier, notre école n'a que le niveau de troisième mais possède quatre classes — 101, 102, 103 et 104. Notre exemple de procédure (sans contrôle d'erreur) ressemble à celui de la Figure 13-1.

```
APPEND BLANK
ERASE
@10,10 SAY 'ENTREZ LES NOMS DES ELEVES' GET NOM
@12,10 SAY 'NUMERO DE LA CLASSE' GET CLASSE
@12,40 SAY 'NIVEAU' GET NIVEAU
READ
RETURN
```

Figure 13-1

Si nous ajoutons un contrôle d'erreur, on obtient la procédure de la Figure 13-2.

```
APPEND BLANK
ERASE
@10,10 SAY 'ENTREZ LES NOMS DES ELEVES' GET NOM
@12,10 SAY 'NUMERO DE LA CLASSE' GET CLASSE
@12,40 SAY 'NIVEAU' GET NIVEAU
READ
IF.NOT.(CLASSE='101'.OR.CLASSE='102'.OR.CLASSE='103'.OR.
  CLASSE='104')
ERASE
@12,10 SAY 'VOUS AVEZ ENTRE ACCIDENTELLEMENT UN NUMERO DE CLASSE
  INCORRECT'
@14,10 SAY 'S'IL VOUS PLAIT ENTREZ LA CLASSE CORRECTE' GET CLASSE
READ
ENDIF
IF.NOT.(NIVEAU='1'.OR.NIVEAU='2'.OR.NIVEAU='3'.OR.NIVEAU='4')
ERASE
@12,10 SAY 'VOUS AVEZ ENTRE ACCIDENTELLEMENT UN NIVEAU INCORRECT'
@14,10 SAY 'S'IL VOUS PLAIT ENTREZ LE NIVEAU CORRECT' GET NIVEAU
READ
ENDIF
RETURN
```

Figure 13-2

Notez la structure des lignes logiques. Elles montrent la façon correcte de manier les opérateurs logiques lorsque vous voulez sélectionner les données parmi plusieurs possibles. Nous avons également la solution suivante :

```
IF.NOT.CLASSE$'101,102,103,104'
IF.NOT.NIVEAU$'1,2,3,4'
```

qui nous permet d'obtenir les mêmes résultats.

Si vous pensez qu'il y a une chance raisonnable de saisir plusieurs fois une donnée incorrecte, même après une « seconde chance », vous pouvez utiliser une boucle « DO » à la place de IF.

```
DO WHILE.NOT.(CLASSE='101'.OR.CLASSE='102'.OR.CLASSE='103'.OR. CLASSE='104')
  ERASE
  @ 12,10 SAY 'VOUS AVEZ ENTRE ACCIDENTELLEMENT UN NUMERO DE CLASSE INCORRECT'
  @ 14,10 SAY 'S'IL VOUS PLAIT ENTREZ LA CLASSE CORRECTE' GET CLASSE
  READ
ENDDO
DO WHILE.NOT.(NIVEAU='1'.OR.NIVEAU='2'.OR.NIVEAU='3'.OR. NIVEAU='4')
  ERASE
  @ 12,10 SAY 'VOUS AVEZ ENTRE ACCIDENTELLEMENT UN NIVEAU INCORRECT'
  @ 14,10 SAY 'S'IL VOUS PLAIT ENTREZ LE NIVEAU CORRECT' GET NIVEAU
  READ
ENDDO
```

Lorsque vous utilisez la boucle DO, vous ne pouvez pas progresser si vous n'avez pas entré la donnée correcte.

Un exemple très parlant d'une rubrique qui nécessite une protection est la rubrique DATE de nos exemples de registre des chèques du chapitre précédent. Dans nos exemples, nous saisissions une date à chaque fois que nous avions un chèque ou un dépôt. Cela apparaît maintenant très dangereux, particulièrement si le nombre de chèques à enregistrer est très important. De plus, nous voulions « voir » l'affichage des chèques basé sur la date.

Le fait d'entrer la date moins souvent sera un moyen efficace de réduire le risque d'erreur. Dans cet exemple, nous aurions pu saisir la date une seule fois au moment de la sélection de la procédure B:MENU. Il suffit de stocker la date dans une variable mémoire et, en utilisant la commande REPLACE, d'écrire celle-ci dans la rubrique DATE.

On peut procéder différemment en utilisant la « date » saisie en début de session dBASE II. On peut alors enregistrer cette date dans la rubrique par

```
REPLACE DATE WITH DATE()
```

Lorsque vous enregistrez la date en début de session dBASE II, elle est stockée à l'emplacement mémoire DATE( ). Il s'agit de la même donnée que la « date de la dernière modification » de votre structure de base de données.

DATE( ) occupe un espace de huit caractères comme la rubrique DATE dans notre exemple. Le fait d'utiliser les huit caractères chaque fois que vous saisissez une date est capital. Nous n'avons pas voulu nous attarder sur ce sujet dans le chapitre précédent. Car il y avait beaucoup d'autres choses importantes. Nous allons parler plus en détail des problèmes de date dans ce chapitre.

Pour commencer, même si nous savons que :

2/12/ équivaut à 02/12/

l'ordinateur lui ne le sait pas. Il compare simplement les caractères, position par position.

```
1 2 3 4 5 6 7 8

2 / 1 2
0 2 / 1 2
```

Comme les rubriques caractères sont justifiées à gauche, la comparaison sur ces rubriques commence par le caractère le plus à gauche. La comparaison, caractère par caractère, comme vous pouvez le remarquer, montre qu'il n'y a aucune comparaison possible. Lorsqu'il y a un grand nombre de données, ce type de « problème » peut être mineur. Dans ce cas cependant, vous pourriez souhaiter d'utiliser la date comme clé pour l'affichage des données. Il est alors très important de procéder à la saisie des caractères de façon précise.

S'il existe une quelconque possibilité d'entrer des données incorrectes, et comme la date est une chose importante pour mettre en « ordre » les enregistrements, l'ordinateur pourra vous aider à saisir ces données de façon correcte par différents

moyens. L'une de ces possibilités est l'utilisation de la commande PICTURE de dBASE II. A la place de l'instruction :

```
@ 10,15 SAY 'ENTREZ LA DATE' GET DATE
```

utilisez :

```
@ 10,15 SAY 'ENTREZ LA DATE' GET DATE PICTURE '99/99/99'
```

Lorsque l'ordinateur exécute cette procédure, il affiche :

```
ENTREZ LA DATE :   /  /   :
```

Les séparateurs « / » sont fixés dans l'affichage. Lorsque vous saisissez les nombres d'une date, l'ordinateur saute les /. Comme avantage supplémentaire, l'ordinateur accepte uniquement les nombres dans les espaces occupés par les 9. Ceci n'élimine pas complètement les problèmes, mais cela peut rendre service, et permet une saisie de données plus facile.

Comment peut-on s'assurer que la date est toujours saisie de manière correcte, avec ou sans la commande PICTURE ? On peut y parvenir par l'astuce d'une procédure utilisant la position des caractères dans la rubrique. Nous avons approché cette idée dans les derniers chapitres. Voyons maintenant cela en détail. Ce n'est pas difficile et c'est capital.

Le troisième et le sixième caractères sont supposés être des /. Nous pouvons le garantir par l'utilisation de la commande PICTURE. Mais même sans PICTURE, nous pouvons placer la date comme nous désirons qu'elle soit.

1	2	3	4	5	6	7	8	
								elle devrait être
j	j	/	m	m	/	a	a	
j	/	m	/	a	a			elle pourrait être

Il y a un grand nombre de variétés de saisies possibles pour cette date. La Figure 13-3 présente une procédure qui restitue le bon format après la saisie.

```
IF.NOT.$(DATE,3,1)='/'                (si le troisième caractère n'est
  STORE '0'+DATE TO DATE               pas égal à /, ajoutez un 0 au
ENDIF                                  début de DATE. Ce qui signifie
                                       que "jour" devait etre infé-
                                       rieur à 10).
IF.NOT.$(DATE,6,1)='/'                 (si le sixième caractère n'est
  STORE $(DATE,1,3)+'0'+$(DATE,5,4); pas une /,)
  TO DATE
ENDIF
```

Figure 13-3

Cette simple petite procédure garantira que la date correspond au format que nous souhaitions.

jj/mm/aa

Cette procédure introduit un nouvel élément, du style :

$(DATE,1,3)

Ce qui signifie : « les trois premiers caractères de la rubrique DATE ». Le premier nombre correspond à la position du caractère de départ. Le deuxième nombre au nombre de caractères que l'on veut extraire. Cet exemple montre bien que nous pouvons manipuler des caractères, ce qui s'avère très souvent utile.

Supposons maintenant que nous voulions indexer le contenu de notre base de données de registre des chèques sur la date. Lorsque nous classons les éléments par date, nous utilisons :

aa/mm/jj

Nous n'y faisons pas réellement attention, mais c'est le seul moyen pour trier les éléments sur la date. Nous utilisons plus souvent :

mm/jj/aa

pour permettre à l'ordinateur de modifier la position des éléments de la date stockée dans la rubrique, nous modifierons les données par :

INDEX ON $(DATE,7,2)+$(DATE,1,5) TO B:DATE

Le fichier index B :DATE nous permet d'accéder aux enregistrements par ordre de date.

Nous avons eu besoin de manipuler la date puisque notre façon de lire les dates n'est pas réellement compatible avec celle utilisée par l'ordinateur. Celui-ci « classe » les chaînes de caractères en commençant par la position caractère la plus à gauche. Les trois dates :

04/22/
03/21/
01/22/

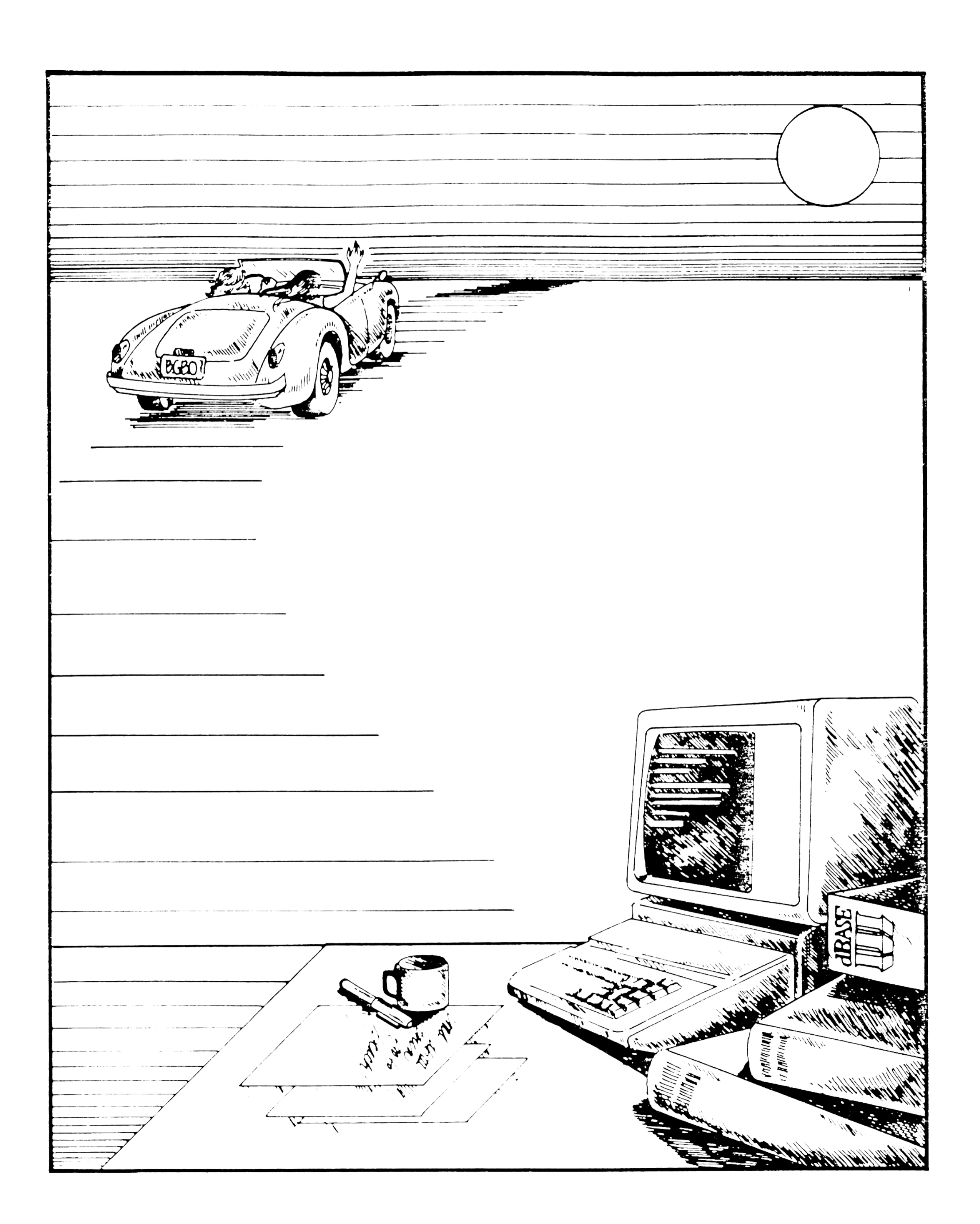
auraient pu être classées comme suit :

01/22/
03/21/
04/22/

si l'on n'avait pas utilisé la procédure décrite précédemment. L'instruction d'indexation que nous venons de voir aurait placé les dates exactement comme suit :

04/22/
01/22/
03/21/

L'ordinateur travaille toujours de façon précise et littérale. Il ne peut travailler qu'avec des contenus comparables. Notre façon de voir est associative, nous savons par exemple que Georges Martin et Monsieur Martin georges sont la même personne. L'ordinateur ne le sait pas. Il travaille par comparaison. Et il y a des moments où nous pouvons être frustrés par cette façon de prendre les choses. Cela signifie cependant que l'on peut prévoir ce que va faire l'ordinateur. Les règles sont fixes et précises. Une fois que nous avons accepté ces règles, nous pouvons obtenir une aide profitable de la part de l'ordinateur.


dBASE

CHAPITRE XIV

DOCUMENTS IMPRIMES SPECIAUX

A l'aide de procédures, vous pouvez préparer des documents imprimés spéciaux dans presque tous les formats que vous pourriez souhaiter. Les procédures servent à préparer des documents imprimés ne pouvant pas être obtenus normalement à partir de l'éditeur de format du SGBD (REPORT). C'est un tout petit peu plus difficile. Le résultat par contre récompensera cet effort puisqu'il sortira le document exactement au format que vous attendez.

Pour illustrer l'usage d'une procédure de préparation de documents spéciaux, nous étudierons un exemple d'un état typique ne pouvant pas être facilement produit par l'éditeur de format standard.

Une école élémentaire utilise un micro-ordinateur avec un système de base de données pour l'enregistrement des élèves. Au début de l'année scolaire, et périodiquement, il est souhaitable de fournir des copies des listes de classes pour les membres de l'administration tels que la bibliothécaire, la secrétaire de l'école et le professeur. La base de données scolaire contient entre autres, des rubriques qui enregistrent chaque progrès de l'élève en lecture et en mathématiques, le numéro de la classe, le niveau et le nom du professeur, ainsi que l'indication si l'élève est redoublant dans son niveau. Les parties essentielles de la structure de cette base de données sont présentées à la Figure 14-1.

NOM DE RUBRIQUE	TYPE DE RUBRIQUE	TAILLE
NOM	C	30
CLASSE	C	3
NIVEAU	C	1
PROFESSEUR	C	15
REDOUBLANT	L	1
LECTURE	N	2
MATH	N	2
SEXE	C	1

Figure 14-1. Structure de la base de données Elèves (partielle)

L'école désire imprimer et distribuer des résultats par classe ressemblant à ceux de la Figure 14-2.

ECOLE SPECIALE

PROFESSEUR : MARTINOT 9 SEPTEMBRE 1985
CLASSE : 101
NIVEAU : 6

RESULTAT DE CLASSE POUR L'ANNEE SCOLAIRE 1985/1986

NOM	LECTURE	MATH	
Alan, Antoine	21	88	
Anerson, Anne	43	29	
Areins, Georgette	87	29	REDOUBLANT
.			
.			
Zimmerman, Robert	34	19	

TAILLE DE LA CLASSE : 28
GARCONS : 14
FILLES : 14

MOYENNE EN LECTURE : 43.32
MOYENNE EN MATHS : 62.67

Figure 14-2. Exemple d'un document imprimé spécial

Cet exemple particulier ne peut pas être préparé (dans ce format) en utilisant l'éditeur de format standard de dBASE II. Le plus gros du rapport peut être géré par REPORT bien entendu. Par contre, les mentions spéciales :

PROFESSEUR
CLASSE
NIVEAU
TAILLE DE LA CLASSE
GARCONS
FILLES
MOYENNE EN LECTURE
MOYENNE EN MATHS

ne peuvent pas se conformer à ce format.

```
SET TALK OFF
USE B:ECOLE
INDEX ON NIVEAU+CLASSE+NOM TO B:RESULTAT
USE B:ECOLE INDEX B:RESULTAT
ACCEPT 'ENTREZ LA DATE' TO DATE
SET PRINT ON
DO WHILE.NOT.EOF
?
?
?
?
?
?'                ECOLE SPECIALE'
?'
?'                PROFESSEUR : ' ,PROFESSEUR, '' , DATE
?'                CLASSE : ',CLASSE
?'                NIVEAU : ',NIVEAU
?
```

```
 ?'            RESULTAT DE CLASSE POUR L'ANNEE SCOLAIRE 1985/1986 '
 ?'
 ?              ______________________________________________
 ?'                  NOM           LECTURE          MATHS'
 ?'             ______________________________________________
 ?
 STORE CLASSE TO MCLASSE
 STORE NIVEAU TO MNIVEAU
 STORE 0 TO GARCONS,FILLES,XLECTURE,XMATHS
DO WHILE CLASSE=MCLASSE.AND.NIVEAU=MNIVEAU.AND..NOT.EOF
 STORE ' ' TO RED
  IF REDOUBLANT
    STORE 'REDOUBLANT' TO RED
  ENDIF
 ?'         ',NOM,'         ',LECTURE,'           ',MATHS,'      ',RED
 IF SEXE='M'
    STORE GARCONS+1 TO GARCONS
  else
    STORE FILLES+1 TO FILLES
 ENDIF
 STORE XLECTURE+LECTURE TO XLECTURE
 STORE XMATHS+MATHS TO XMATHS
 SKIP
ENDDO
 ?'       TAILLE DE LA CLASSE :',STR(GARCONS+FILLES,2)
 ?'       GARCONS :            ',STR(GARCONS,2)
 ?'       FILLES :             ',STR(FILLES,2)
 ?
 ?'       MOYENNE EN LECTURE : ',STR(XLECTURE/(GARCONS+FILLES),6,2)
 ?'       MOYENNE EN MATHS :   ',STR(XMATHS/(GARCONS+FILLES),6,2)
 EJECT
ENDDO
SET PRINT OFF
SET TALK ON
CANCEL
```

Figure 14-3.

Cette procédure fournira un jeu de documents imprimés avec chaque résultat de classe imprimé sur une page séparée. Ce rapport est entièrement à la mesure des besoins et des souhaits des personnes qui l'ont demandé. Dans les paragraphes qui suivent, nous étudierons en détail chaque étape de cette procédure.

Pour ce faire, nous commencerons par les éléments de base de cette procédure. Puis nous ajouterons de nouveaux éléments et continuerons à en ajouter jusqu'à ce que nous arrivions à la procédure finale. A chaque passage, nous ajouterons la nouvelle ligne (ou celles qui sont modifiées) en caractères gras. L'élément le plus important est une boucle DO qui produit un listing continu de tous les élèves de l'école (ce n'est rien moins qu'une version sous forme de procédure de la commande LIST).

```
SET TALK OFF
DO WHILE.NOT.EOF
  DISPLAY
  SKIP
ENDDO
SET TALK ON
CANCEL
```

Figure 14-4. Version sous forme de procédure de la commande LIST de dBASE II

SET TALK OFF/ON

Presque toutes les procédures commencent et se terminent par cette commande. Lorsque l'on travaille à partir du clavier, il est souhaitable que l'ordinateur vous réponde (TALK) chaque fois que vous expérimentez une commande. Ce n'est pas souhaitable lors de l'utilisation de procédures. Vous pouvez désirez que l'ordinateur ne « parle » uniquement qu'à votre demande. Dans ce cas, nous voulons voir le contenu des enregistrements (DISPLAY) mais nous ne voulons pas obtenir en écho, le numéro d'enregistrement à partir de la commande SKIP.

SKIP

Cette commande avancera la base de données d'un enregistrement chaque fois qu'elle sera utilisée. Si nous n'avions pas SET TALK OFF, l'ordinateur afficherait le numéro d'enregistrement à chaque utilisation de SKIP.

Si SKIP est employée avec une base de données non indexée, les enregistrements avanceront dans l'ordre des numéros d'enregistrement. Si la base de donnée utilisée est indexée, les enregistrements avanceront dans leur ordre « logique ».

```
DO WHILE.NOT.EOF
```

Indique à l'ordinateur de répéter les commandes DISPLAY et SKIP jusqu'à ce que l'on ait atteint la fin du fichier de base de données.

Dans l'étape suivante d'explication de la procédure générant un document spécial, nous voulons :

- Indiquer à l'ordinateur quel fichier de base de données utiliser
- INDEXer la base de données par classe
- Indiquer à l'ordinateur d'utiliser la base de données indexée
- Afficher seulement les rubriques que nous souhaitons.

```
SET TALK OFF
USE B:ECOLE
INDEX ON NIVEAU+CLASSE+NOM TO B:RESULTAT
USE B:ECOLE INDEX B:RESULTAT
DO WHILE.NOT.EOF
  ? NOM,LECTURE,MATHS
  SKIP
ENDDO
SET TALK ON
CANCEL
```

Figure 14-5. Procédure modifiée de la commande LIST de dBASE II

Cette procédure produit maintenant l'affichage écran avec les élèves présentés dans l'ordre alphabétique par niveau et par classe. La commande DISPLAY a été remplacée par le symbole ?. Cela a pour effet d'empêcher l'affichage du numéro d'enregistrement et nécessite moins de frappe que DISPLAY OFF qui aurait donné le même résultat. Seul le contenu des rubriques NOM, LECTURE et MATHS seront affichés.

Aux étapes suivantes, nous ajouterons quelques commandes pour produire un simple document imprimé des résultats scolaires. A ce stade du développement de cette procédure, le document imprimé sera très sommaire. Chaque résultat scolaire consistera en une liste de noms d'élèves, de notes en lecture et en maths, commençant à l'extrémité supérieure de la page. Il n'y aura pas de marge gauche.

```
SET TALK OFF
USE B:ECOLE
INDEX ON NIVEAU+CLASSE+NOM TO B:RESULTAT
USE B:ECOLE INDEX B:RESULTAT
SET PRINT ON
DO WHILE.NOT.EOF
  STORE CLASSE TO MCLASSE
  STORE NIVEAU TO MNIVEAU
  DO WHILE CLASSE=MCLASSE.AND.NIVEAU='MNIVEAU'.AND..NOT.EOF
  ? NOM,LECTURE,MATHS
  SKIP
  ENDDO
  EJECT
ENDDO
SET PRINT OFF
SET TALK ON
CANCEL
```

Figure 14-6. Version élémentaire d'une procédure de résultat scolaire

SET PRINT ON/OFF

Cela a pour effet de mettre en ou hors fonction l'imprimante. Vous devez habituellement mettre en fonction l'imprimante après SET TALK OFF. De la même façon, mettez-la hors fonction avant SET TALK ON.

```
STORE CLASSE TO MCLASSE
STORE NIVEAU TO MNIVEAU
```

Ces deux commandes vous permettent de mettre en place la boucle DO pour lister chaque classe séparément. Elles permettent à l'ordinateur d'établir automatiquement le début et la fin d'un regroupement par classe.

```
DO WHILE CLASSE=MCLASSE.AND.NIVEAU=MNIVEAU.AND..NOT.EOF
```

Ici, nous avons une boucle DO dans une boucle DO. La boucle intérieure est entièrement contenue dans la boucle extérieure. Tant que chaque enregistrement obéit aux conditions suivantes :

- Le numéro de la classe est le même que MCLASSE
- Le niveau est identique à MNIVEAU
- La fin du fichier n'a pas été rencontrée.

La procédure continue à imprimer chaque nom d'élève, son niveau en lecture, et son niveau en maths. Notez les doubles points entre AND et NOT. C'est la manière correcte pour entrer une condition.

```
EJECT
```

EJECT provoque l'avance du papier de l'imprimante sur la page suivante.

Cette procédure est relativement directe. La boucle extérieure permet le listage de l'ensemble de la base de données. Nous entrons pour la première fois dans la boucle extérieure (DO WHILE.NOT.EOF), la classe et le niveau du premier enregistrement sont stockés dans les variables mémoire MCLASSE ET MNIVEAU. Ensuite, nous entrons dans la boucle intérieure. Cette boucle sera répétée jusqu'à ce que la base de données ait avancée vers un enregistrement dont la CLASSE et le NIVEAU ne sont pas égaux au contenu des variables mémoire MCLASSE et MNIVEAU. Si c'est le cas, il se produit un envoi de papier sur l'imprimante et nous revenons au début de la boucle extérieure (nous rangeons la nouvelle classe et le niveau dans les variables mémoire et nous continuons). Si nous rencontrons la marque de fin de fichier, nous éteignons l'imprimante et nous avons terminé. Notez que la boucle intérieure a également .NOT.EOF comme élément de sa condition.

A ce stade, nous sommes prêt à ajouter des commandes qui exécutent des actions nécessaires pour :

- Compter les garçons
- Compter les filles
- Cumuler le contenu des rubriques lecture et maths pour chaque classe.

```
SET TALK OFF
USE B:ECOLE
INDEX ON NIVEAU+CLASSE+NOM TO B:RESULTAT
USE B:ECOLE INDEX B:RESULTAT
SET PRINT ON
DO WHILE.NOT.EOF
  STORE CLASSE TO MCLASSE
  STORE NIVEAU TO MNIVEAU
  STORE 0 TO GARCONS,FILLES,XLECTURE,XMATHS
DO WHILE CLASSE=MCLASSE.AND.NIVEAU='MNIVEAU'.AND..NOT.EOF
 ?'          'NOM,LECTURE,MATHS
 IF SEXE='M'
   STORE GARCONS+1 TO GARCONS
 ELSE
   STORE FILLES+1 TO FILLES
 ENDIF
 STORE XLECTURE+LECTURE TO XLECTURE
 STORE XMATHS+MATHS TO XMATHS
 SKIP
ENDDO
 EJECT
ENDDO
SET PRINT OFF
SET TALK ON
CANCEL
```

Figure 14-7.

```
STORE 0 TO GARCONS,FILLES,XLECTURE,XMATHS
```

Cette commande crée quatre variables mémoire :

GARCONS
FILLES
XLECTURE
XMATHS

et met leurs valeurs initiales à zéro.

```
IF SEXE='M'
  STORE GARCONS+1 TO GARCONS
ELSE
  STORE FILLES+1 TO FILLES
ENDIF
```

Dans la boucle intérieure, nous incrémentons de un la valeur de GARCONS si le contenu de SEXE est M. Autrement, nous incrémentons FILLES de un (incrémenter signifie ajouter un à la valeur initiale).

```
STORE XLECTURE+LECTURE TO XLECTURE
STORE XMATHS+MATHS TO XMATHS
```

De plus, nous ajoutons le contenu de la rubrique lecture à la variable mémoire XLECTURE et celui de la rubrique maths à la variable mémoire XMATHS.

A ce stade, il nous faut mettre en place une procédure capable d'imprimer les informations spécialement formatées au bas de chaque résultat scolaire.

TAILLE DE LA CLASSE
GARCONS
FILLES
MOYENNE EN LECTURE
MOYENNE EN MATHS

C'est le résultat des commandes suivantes :

```
?
?'          TAILLE DE LA CLASSE :',STR(GARCONS+FILLES,2)
?'          GARCONS :             ',STR(GARCONS,2)
?'          FILLES :              ',STR(FILLES,2)
?
?'          MOYENNE EN LECTURE :',STR(XLECTURE/(GARCONS+FILLES),6,2)
?'          MOYENNE EN MATHS :  ',STR(XMATHS/(GARCONS+FILLES),6,2)
```

Le point d'interrogation (?), lorsqu'il est utilisé seul, produit une ligne vierge sur l'écran et/ou sur l'imprimante. La ligne de commande

```
?'          TAILLE DE LA CLASSE :',STR(GARCONS+FILLES,2)
```

fournit une ligne imprimée qui ressemble à celle montrée dans l'exemple. Les espaces blancs entre les apostrophes sont présents afin de constituer une marge gauche. Sans ces espaces, le mot CLASSE commencerait à l'extrémité gauche du papier.

```
STR(GARCONS+FILLES,2)
```

Cette partie de la commande prend la somme des variables mémoire GARCONS et FILLES et imprime le résultat sous forme de chaîne de caractères à deux positions. Dans ce cas, nous savons que le résultat ne peut pas contenir plus de deux caractères. Dans le cas contraire, il suffirait simplement d'utiliser la ligne de commande :

```
?'          TAILLE DE LA CLASSE :',GARCONS+FILLES
```

La somme serait imprimée sur une rubrique à dix positions. En plaçant 8 espaces vierges, entre le texte « TAILLE DE LA CLASSE » et la somme imprimée.

```
?'          MOYENNE EN LECTURE :',STR(XLECTURE/(GARCONS+FILLES),6,2)
```

Ici, nous trouvons une autre version de cette commande avec « ,2 » en plus, qui indique à l'ordinateur que nous voulons l'affichage de deux décimales.

Maintenant nous sommes prêt pour ajouter les commandes qui impriment l'en-tête de la page et formatent correctement le document.

```
SET TALK OFF
USE B:ECOLE
INDEX ON NIVEAU+CLASSE+NOM TO B:RESULTAT
USE B:ECOLE INDEX B:RESULTAT
ACCEPT 'ENTREZ LA DATE' TO DATE
SET PRINT ON
DO WHILE.NOT.EOF
  ?
  ?
  ?
  ?
  ?
  ?'                              ECOLE SPECIALE
  ?'
  ?'               PROFESSEUR : ',PROFESSEUR,'             ',DATE
  ?'               CLASSE :     ',CLASSE
  ?'               NIVEAU :     ',NIVEAU
  ?
  ?'               RESULTAT DE CLASSE POUR L'ANNEE SCOLAIRE 1985/1986'
  ?'               _________________________________________________'
  ?
  ?'                   NOM            LECTURE            MATHS'
  ?'               _________________________________________________'
  ?
  STORE CLASSE TO MCLASSE
  STORE NIVEAU TO MNIVEAU
  STORE 0 TO GARCONS,FILLES,XLECTURE,XMATHS
DO WHILE CLASSE=MCLASSE.AND.NIVEAU=MNIVEAU.AND..NOT.EOF
 STORE ' ' TO RED
  IF REDOUBLANT
    STORE 'REDOUBLANT' TO RED
  ENDIF
```

```
  ?'          ',NOM,LECTURE,MATHS,'     ',RED
  IF SEXE='M'
     STORE GARCONS+1 TO GARCONS
   else
     STORE FILLES+1 TO FILLES
  ENDIF
  STORE XLECTURE+LECTURE TO XLECTURE
  STORE XMATHS+MATHS TO XMATHS
  SKIP
 ENDDO
  ?
  ?'       TAILLE DE LA CLASSE :',STR(GARCONS+FILLES,2)
  ?'       GARCONS :            ',STR(GARCONS,2)
  ?'       FILLES :             ',STR(FILLES,2)
  ?
  ?'       MOYENNE EN LECTURE : ',STR(XLECTURE/(GARCONS+FILLES),6,2)
  ?'       MOYENNE EN MATHS :   ',STR(XMATHS/(GARCONS+FILLES),6,2)
  EJECT
 ENDDO
 SET PRINT OFF
 SET TALK ON
 CANCEL
```

Figure 14-8.

```
ACCEPT 'ENTREZ LA DATE' TO DATE
```

Cette commande nous permet d'entrer la date dans le format que nous souhaitons. Elle déclenche le message :

```
ENTREZ LA DATE :
```

affiché sur l'écran. L'ordinateur fait une pause jusqu'à ce que vous ayez entré la date et appuyé sur RETURN. La date saisie sera rangée dans la variable mémoire DATE. Avant l'arrivée de cette commande, bien avant la boucle DO, il vous faudra entrer la date une seule fois pour l'ensemble des résultats scolaires.

Notez que la commande intervient avant l'allumage de l'imprimante.

En-tête de page

L'en-tête de page est imprimée au moyen de la commande ?. Un ? utilisé seul donne une ligne vierge imprimée et/ou affichée. Le texte encadré par des apostrophes (délimiteurs) sera imprimé comme dans l'exemple. (Les délimiteurs ne sont pas imprimés). Le contenu des variables mémoire ou des rubriques de données seront imprimés lorsque le nom de rubrique ou le nom de la variable est utilisé comme suit :

```
?          ',NOM,'          ',LECTURE,'          ',MATHS,'          ',RED
```

Ce qui nous donne le format d'affichage de base pour chaque enregistrement imprimé. Les espaces blancs sont utilisés pour positionner les colonnes de données sur la page. RED est une variable mémoire indiquant si l'élève est redoublant ou non. La variable doit être créée et contenir l'information correcte avant cette ligne de commande.

```
STORE' ' TO RED
IF REDOUBLANT
  STORE 'REDOUBLANT' TO RED
ENDIF
```

Ces commandes nous montrent comment mettre en place la variable mémoire RED pour un usage ultérieur. Ce processus doit être répété pour chaque donnée d'enregistrement.

Comme vous avez pu l'observer, il n'y a aucune difficulté pour préparer une procédure qui confectionne un document imprimé spécial. Il vous suffit simplement de faire preuve d'attention et de faire les choses méthodiquement. Chaque étape du processus doit être saisie dans l'ordinateur. Il est conseillé d'ajouter la commande SET PRINT ON après s'être assuré que la procédure marche convenablement. Ceci vous permet de contrôler votre procédure sur le terminal sans gâcher de papier.

CINQUIEME PARTIE

Maintenant que nous sommes des experts en programmation (et vous pensiez que la programmation était uniquement l'affaire des informaticiens !), la cinquième partie illustre la manière dont fonctionne une application de base de données dans une situation professionnelle.

Notre exemple de « Boutique vidéo » nous permet d'appliquer nos méthodes de base de données aux nombreuses procédures que l'on peut rencontrer lorsque l'on fait tourner une affaire. Notre système de gestion de base de données va pouvoir fournir une grande variété de services, depuis les mailings à la gestion de stock en passant par l'enregistrement d'opérations commerciales spécialisées.

CHAPITRE XV

UTILISATION PROFESSIONNELLE

L'objectif principal d'un micro-ordinateur et d'une gestion de base de données est de vous aider. Particulièrement, dans l'activité de gestion d'une petite affaire. D'après un récent rapport, il existe plus de trois millions d'entreprises dont le chiffre d'affaires est inférieur à 5 millions de dollars employant moins de dix personnes.

Une boutique vidéo est un exemple privilégié de ce type d'activité. La boutique vidéo spécialisée dans la vente et la location d'appareils de télévision, principalement des magnétoscopes (VCR), des cassettes de films vidéo, et des accessoires. En plus de la vente et de la location de marchandises, ces boutiques financent souvent des « vidéo clubs » qui donnent la possibilité à ses membres d'obtenir des tarifs réduits sur les locations et les matériels.

Un système de gestion de base de données peut s'appliquer à de nombreux secteurs d'activité :

- PAIE ET COMPTABILITE
- DECLARATION FISCALE
- GESTION DE STOCK
- CHARGES IMMOBILIERES
- MAILINGS
- REGISTRE JOURNALIER DE CAISSE
- ENREGISTREMENT D'OPERATIONS COMMERCIALES

Cette liste n'est certainement pas exhaustive. Cependant, elle est vraiment représentative des applications professionnelles prises en charge par un SGBD. Un tel système aide à mieux gérer les opérations avec moins d'efforts. Nous allons étudier une boutique vidéo car c'est un exemple représentatif pour l'explication de certains concepts. Cet exemple ne fait pas appel à trop de termes spécialisés.

Les locations d'une boutique vidéo

La location de cassettes de films pré-enregistrés est la part la plus importante de l'activité d'une boutique vidéo. La boutique acquiert ou loue des films pour les louer ensuite. Chaque bande représente un investissement et se trouve à un emplacement particulier. Il est important que le propriétaire sache quels sont les films qui marchent bien en location et ceux qui marchent moins bien. Un film qui ne tourne pas assez souvent est soit mis « à la vente » soit retourné à son propriétaire. Il est également important de contrôler le nombre de locations d'un film populaire. Les clients peuvent être mécontents de bandes usagées ou de films de mauvaise qualité. Un système de gestion de base de données peut suivre une telle activité et permet de réduire le temps nécessaire pour contrôler le nombre de passages en location d'une bande.

Beaucoup de boutiques vidéo constituent un « vidéo club ». Ce qui implique pour ses membres le paiement d'une cotisation forfaitaire donnant droit à des réductions significatives sur les locations de films ou achat de matériel. Ces « clubs » sont intéressants pour la boutique aussi bien que pour le client. Une liste des membres du club constitue souvent la base d'une liste de mailing commercial. Un système de gestion de base de données peut simplifier la mise à jour d'une liste de mailing et imprimer des étiquettes pour le routage. Cette liste peut constituer également le moyen de relancer les membres dont les cotisations sont arrivées à expiration.

Il y a un certain nombre d'autres niveaux de tâches administratives variées avec ou sans l'ordinateur qui sont faites à l'aide d'un crayon et d'un papier. Un système d'ordinateur doit pouvoir exécuter des services supplémentaires pour réduire la quantité de travail et devenir rentable.

La réception de nouvelles marchandises nécessite la mise à jour du stock et leur enregistrement en comptabilité. La vente d'une marchandise implique l'enregistrement de la vente ainsi que la modification des éléments du stock. Un système de base de données s'adapte facilement aux gestions de stock comme à la plupart des routines de gestion administrative et comptable.

« Entreposer » des marchandises, est une tâche tout à fait commune dans les affaires. La boutique obtient des marchandises pour une période (souvent 90 jours) avant que soit dû le paiement de celles-ci. Le paiement est dû immédiatement si la marchandise est vendue précédemment à la date exigible. Une somme est souvent payée chaque mois pour alimenter le compte clients du fournisseur. De telles dispositions nécessitent une gestion soigneuse, une tâche qui peut être aisément prise en charge par un système de gestion de base de données.

Cela ne constitue-t-il pas un bon point de départ pour un système informatique ? Puisqu'une base de données créée avec un papier et un crayon est familière et utile à tout le monde, nous allons commencer par là. Nous remplacerons exactement notre base de données « ancien style » par un système de base de données informatique. Réfléchissons d'abord de façon générale sur le travail que cela implique.

Un certain nombre de choses se produisent chaque jour :

- On reçoit un nouveau stock
- Le stock est vendu
- Des cartes de club sont vendues
- Des films et des matériels sont loués
- Des films et des matériels sont rentrés
- Les opérations de caisse sont balancées

Chacune de ces activités nécessite un travail administratif. Par exemple, le livre de caisse, résumant le travail commercial de la journée et comptabilisant les échanges d'argent, doit être mis à jour quotidiennement. Son format est présenté à la Figure 15-1.

	10F	
	50F	
	100F	
	500F	

15. Total des billets ——————

	10C	
	50C	
	1C	
	2C	
	5C	

16. Total de la monnaie ——————

Figure 15-1

PAGE DU REGISTRE DE CAISSE Jour ——— Mois ———
Année ———
Responsable ————

COMPTER LA CAISSE A L'OUVERTURE

1. Total de départ du tiroir-caisse ——————————— F

V	2. Locations (cassettes films et matériel)	———— F
E N	3. Abonnements (A vie—— Année——— Autre———)	———— F
T	4. Services (Réparation Matériel + installation)	———— F
E S	5. Matériel et Accessoires	———— F
	6. Arrhes	———— F

---

8. Total des Ventes (2+3+4+5+6)
9. Total paiement en espèces (reçus)
10. Total des Ventes (-) les espèces (8 moins 9)

COMPTABILISER LES ENTREES DU REGISTRE
EN FIN DE JOURNEE

11. Espèces (15+16)
12. Chèques (Nb de chèques ————)
13. Cartes de crédit (Visa et cartes bleues)
14. Total caisse du registre (11+12+13)

---

17. TOTAL (1+10)
18. Total du Registre (14)

Différence éventuelle entre (17 et 18)
en moins ——— en plus ———

Total des espèces

Moins les espèces en caisse à l'ouverture
Montant des dépôts pour caution

Les activités résumées sur ce format constituent une large part du travail administratif quotidien d'une journée à la boutique. Nous commençons ici la démonstration d'un support approprié de gestion de base de données. Comme pour tous les processus précédents, la phase 1 est la CONCEPTION de la base.

La conception d'une base de données doit partir d'une compréhension réelle de ce qui doit être accompli. Jusque-là, nous avons identifié plusieurs documents papier utilisés dans les opérations de la boutique.

La duplication de chaque document papier par un « document électronique » est un moyen utile et facile de commencer. Ceci vous permet de construire votre système pièce par pièce. Vous pouvez aisément contrôler les résultats avec vos documents papier, vous avez là un moyen de tester votre système informatique.

Dans notre exemple, l'employé choisit parmi un menu des différents « formats électroniques ». Le menu de départ est montré par l'Ecran 15-1.

```
SYSTEME DE GESTION D'UNE BOUTIQUE VIDEO

          1 - LOCATION DE FILMS

          2 - ABONNEMENTS

          3 - VENTES

          4 - SERVICES

          5 - CAUTIONS

          6 - REGISTRE DE CAISSE

          7 - STOCK

          8 - FIN
```

Ecran 15-1

Un diagramme montrant la relation de ces différents éléments est présenté par la Figure 15-2.

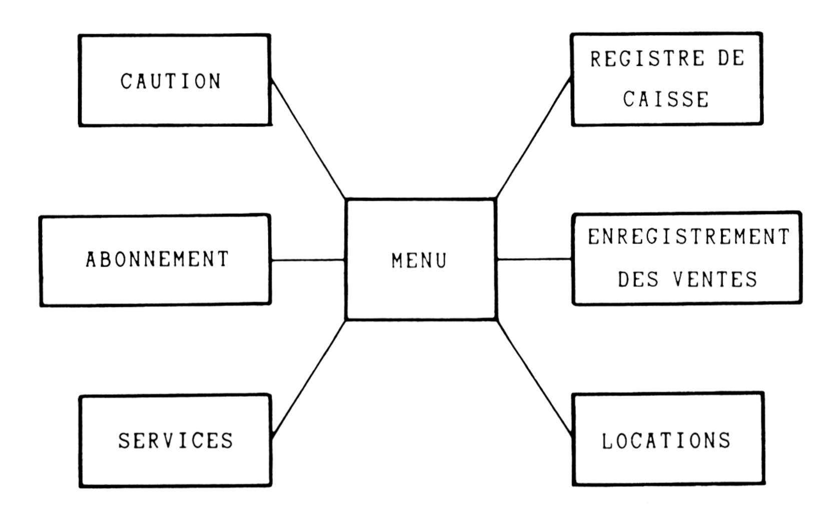


Figure 15-2

Nous convertissons d'abord le « format papier » en un « format électronique ». Nous avons besoin d'une procédure et d'une base de données. Le plan de la base de données — B :REGISTRE — est présenté à la Figure 15-3.

RUBRIQUE	DESCRIPTION	NOM DE RUBRIQUE	TYPE	TAILLE	DECIMALE(S)
1	Date	DATE	C	8	
2	Nom de l'employé	NOM	C	10	
3	Espèces à l'ouverture	ESPECDEP	N	6	2
4	Total des locations	LOCATION	N	7	2
5	Membres abonnés à vie	MEMBREVIE	N	2	
6	Membres abonnés à l'année	MEMBREAN	N	2	
7	Autres membres	AUTRMEMBR	N	2	
8	Encaissements des abonnés	ENCABONN	N	7	2
9	Prestations de service	SERVICES	N	7	2
10	Ventes Hors TVA	VENTSHT	N	8	2
11	Ventes TTC	VENTESTTC	N	8	2
12	Cautions	CAUTIONS	N	7	2
13	Total paiements en espèces	TOTESPECES	N	7	2
14	Espèces à la fermeture	ESPECFERM	N	8	2
15	Nombre total de chèques	NBRCHEQ	N	3	
16	Valeurs en chèques	TOTCHEQ	N	8	2
17	Cartes de crédit	CARTCREDIT	N	6	2
18	Caisse pour le lendemain	INITCAISSE	N	8	2
19	Montant dépôts à la banque	DEPOTBANQ	N	8	2

Figure 15-3. Plan du registre de Caisse

Une feuille du registre de caisse équivaut à un enregistrement dans la base de données registre de caisse. De nombreux éléments de cette page de registre n'ont pas de rubrique correspondante sur le format électronique puiqu'ils sont le résultat de calculs à partir d'éléments contenus dans la base de données.

La Figure 15-4 montre un exemple de procédure simple permettant à l'employé de remplir les rubriques. Les lignes débutant avec NOTE sont ignorées par l'ordinateur. Ce sont des explications pour l'opérateur qui décrivent l'utilité et le fonctionnement de certaines parties de la procédure.

Dans cette procédure particulière, l'employé entre à la main quotidiennement toutes les informations de chaque opération, exactement comme il le ferait sur une page du registre de caisse. L'ordinateur exécute cependant tous les calculs nécessaires.

```
USE B:REGISTRE
SET TALK OFF
GO BOTTOM
(Après avoir mis en place la base de données, nous devons ajouter
au moins un enregistrement vide ou disposer d'un certain nombre
d'enregistrements corrects dans la base pour que ce programme puisse
s'exécuter)
IF.NOT.DATE=DATE()
APPEND BLANK
REPLACE DATE WITH DATE()
ENDIF
ERASE
@0,10 SAY 'REGISTRE DE CAISSE DU' GET DATE
@2,10 SAY 'Nom de l'employé' GET NOM
@3,10 SAY 'Espèces à l'ouverture' GET ESPECEDEP
@4,10 SAY 'Total des locations' GET LOCATION
@5,10 SAY 'Membres abonnés à vie' GET MEMBREVIE
@6,10 SAY 'Membres abonnés à l'année' GET MEMBREAN
@7,10 SAY 'Autres membres' GET AUTRMEMBR
@8,10 SAY 'Encaissements des abonnés' GET ENCABONN
@9,10 SAY 'Prestations de service' GET SERVICES
@10,10 SAY 'Ventes Hors TVA' GET VENTSHT
@11,10 SAY 'Ventes TTC' GET VENTESTTC
@12,10 SAY 'Cautions' GET CAUTIONS
@13,10 SAY 'Total paiements en espèces' GET TOTESPECES
@14,10 SAY 'Espèces à la fermeture' GET ESPECFERM
@15,10 SAY 'Nombre total de chèques' GET NBRCHEQ
@16,10 SAY 'Valeurs en chèques' GET TOTCHEQ
@17,10 SAY 'Cartes de crédit' GET CARTCREDIT
@18,10 SAY 'Caisse pour le lendemain' GET INITCAISSE
@19,10 SAY 'Montant dépôts à la banque' GET DEPOTBANQ
READ
NOTE maintenant nous disposons de toutes les données pour calculer le
total du registre de caisse
```

(suite sur la page suivante)

```
STORE LOCATION+ENCABONN+SERVICES+VENTESHT+VENTESTTC+CAUTIONS
  TO TOTVENTES
STORE TOTVENTES-TOTESPECES+ESPECDEP TO TOTAL
STORE ESPECFERM+TOTCHEQ+CARTCREDIT TO RECETTE
ERASE
@8,10 SAY 'VALEUR DE LA CAISSE A L'OUVERTURE' GET ESPECDEP
@10,10 SAY 'CHIFFRE D'AFFAIRES DE LA JOURNEE' GET TOTVENTES
@12,10 SAY 'MONTANT DES PAIEMENTS EN ESPECES' GET TOTESPECES
@14,10 SAY 'LE CHIFFRE D'AFFAIRES DU REGISTRE DOIT ETRE DE :'
   GET TOTAL
@16,10 SAY 'LE MONTANT DU REGISTRE EST DE :' GET RECETTE
DO CASE
  CASE RECETTE>TOTAL
  STORE RECETTE-TOTAL TO DIFF
  @18,10 SAY 'LE REGISTRE EST EXCEDENTAIRE DE :' GET DIFF
  CASE TOTAL>RECETTE
  STORE TOTAL-RECETTE TO DIFF
  @18,10 SAY 'LE REGISTRE EST DEFICITAIRE DE :' GET DIFF
ENDCASE
@20,5 SAY 'APPUYEZ SUR UNE TOUCHE POUR CONTINUER'
WAIT
SET TALK ON
RETURN
```

Figure 15-4

L'ordinateur peut potentiellement faire beaucoup plus, par exemple vous aider à compter l'argent liquide et à entrer les chèques et les charges comme les exemples de registre de chèques du Chapitre XI.

Nous avons parlé précédemment du fonctionnement d'un « vidéo club » avec leurs abonnés comme faisant partie de cette activité. Lorsqu'il y a constitution d'un membre, l'employé remplit un document contenant :

1. Le nom du membre
2. L'adresse du membre
3. Le numéro de téléphone du membre
4. Le numéro d'abonnement au vidéo club
5. La cotisation d'abonnement
6. Le type d'abonnement (à vie, annuel, etc.)
7. La date d'abonnement

Ce format constitue un simple enregistrement dans notre fichier de base de données abonnés, appelé B :ABONNES dans cet exemple. Le plan de la base de données B :ABONNES est montré à la Figure 15-5.

RUBRIQUE	DESCRIPTION	NOM DE RUBRIQUE	TYPE	TAILLE	DECIMALE(S)
1	Date	DATE	C	8	
2	Nom du membre	MNOM	C	30	
3	Numéro et rue	NORUE	C	20	
4	Ville	VILLE	C	20	
5	Code postal	CP	C	5	
6	Numéro de téléphone	TEL	C	8	
7	Type d'abonnement	TYPE	C	1	
8	Cotisations	COTISATION	N	6	2
9	Numéro d'abonnement	NOMEMBRE	C	8	

Figure 15-5. Plan d'un fichier d'abonnement au vidéo club

La procédure d'« enregistrement » de nouveaux membres est extrêmement simple. Appelée B :ABONNES, elle est présentée à la Figure 15-6.

```
USE B:ABONNES
SET TALK OFF
ERASE
GO BOTTOM
STORE VAL(NOMEMBRE)+1 TO M1
APPEND BLANK
REPLACE NOMEMBRE WITH STR(M1,8),DATE WITH DATE()
@ 3,10 SAY 'FICHIER DES ABONNES'
@ 8,10 SAY 'NUMERO DE MEMBRE' GET NOMEMBRE
@ 8,40 SAY 'DATE' GET DATE
CLEAR GETS
@10,10 SAY 'Nom de l'abonné' GET MNOM
@12,10 SAY 'Numéro et rue' GET NORUE
@14,10 SAY 'Ville' GET VILLE
@16,10 SAY 'Code postal' GET CP
@18,10 SAY 'Numéro de téléphone' GET TEL
@20,10 SAY 'Type d'abonnement' (V - à vie A - annuel
   R - renouvellement)' GET TYPE
READ
DO CASE
  CASE TYPE='V'
    REPLACE COTISATION WITH 3.000,00
  CASE TYPE='A'
    REPLACE COTISATION WITH 1.000,00
  CASE TYPE='R'
    REPLACE COTISATION WITH 300,00
ENDCASE
CLEAR GETS
@22,10 SAY 'Montant de la cotisation' GET COTISATION
READ
SET TALK ON
RETURN
```

Figure 15-6

En établissant cette procédure, nous sommes parvenus à un point important. Nous pouvons lier le formulaire d'abonnement (après s'être assuré qu'il fonctionne correctement) avec le « registre de caisse ». Cette opération de liaison est particulièrement facile puisque l'ajout de nouveaux abonnés et la saisie de données dans le registre de caisse sont faits manuellement. Maintenant, la constitution d'un abonnement peut mettre à jour automatiquement le registre de caisse.

Pour pouvoir obtenir ce résultat, il nous suffira d'insérer à la procédure B :ABONNES, Figure 15-6, la procédure de la Figure 15-7 entre les instructions READ et SET TALK ON.

```
READ (IL S'AGIT DU READ QUE L'ON PEUT VOIR A LA FIGURE 15-6)
STORE TYPE TO A1
STORE COTISATION TO A2
USE B:REGISTRE
GO BOTTOM
IF.NOT.DATE=DATE()
  APPEND BLANK
  REPLACE DATE WITH DATE()
ENDIF
DO CASE
  CASE A1='V'
    REPLACE MEMBREVIE WITH MEMBREVIE+1
    REPLACE ENCABONN WITH ENCABONN+A2
  CASE A1='A'
    REPLACE MEMBREAN WITH MEMBREAN+1
    REPLACE ENCABONN WITH ENCABONN+A2
  CASE A1='R'
    REPLACE AUTRMEMBR WITH AUTRMEMBR+1
    REPLACE ENCABONN WITH ENCABONN+A2
ENDCASE
SET TALK ON (CETTE INSTRUCTION VIENT DE LA FIGURE 14-6)
```

Figure 15-7

Dans cet exemple d'abonnement, nous avons ajouté toutes les informations à la base de données des abonnements B :ABONNES. En même temps, sans effort, la mise à jour peut se faire sur le registre de caisse.

La majeure partie du travail d'une boutique vidéo est la location de cassettes de films vidéo. Ces films sont soit des acquisitions ou des locations. Lorsqu'un film est loué à un client, on lui fait remplir un imprimé qu'il signe. L'imprimé contient :

1. Le nom du client
2. L'adresse du client
3. Le numéro de téléphone du client
4. Le numéro de permis de conduire
   ou
   Le numéro d'abonnement au vidéo club
5. La cotisation de location
6. Le montant de la caution (s'il y en a)
   (les membres du club n'ont pas besoin de déposer une caution)
7. Le nombre de films loués
8. Les noms des films loués
9. La date de location
10. La date de retour

Lorsque les films sont rendus, l'imprimé est « annulé ». La boutique utilise cet imprimé pour garder la trace de :

- La transaction de caisse
- Le lieu où se trouvent les films
- Le nombre de passages en location d'un film

Au premier abord, la manière directe de gérer les affaires de location sur notre ordinateur serait d'avoir un enregistrement pour chaque transaction. Le seul problème avec cette méthode concerne l'élément 8. Comment savoir à l'avance l'espace nécessaire pour la rubrique qui gèrera le nom des films. Si nous travaillons sur papier, nous pouvons adopter une petite écriture. Mais l'ordinateur ne sait pas faire ce genre de chose.

Le seul moyen pour résoudre ce problème est d'utiliser plusieurs fichiers pour constituer la base de données de location. Le premier contiendra toutes les informations nécessaires excepté les titres des films. Le titre de chaque film loué deviendra l'enregistrement d'une deuxième base de données. Les deux bases de données seront liées ensemble avec un identificateur de transaction. Cet identificateur de transaction doit être unique pour chaque transaction. Il peut se trouver sur une

ou plusieurs rubriques. Dans cet exemple, le seul identificateur de transaction pourrait être fourni par le numéro de permis de conduire du client (ou le numéro d'abonnement) et la date.

Le rapprochement entre ces deux fichiers de base de données est montré de manière plus graphique à la Figure 15-8. Les informations contenues dans ces deux fichiers sont présentées côte à côte dans des colonnes.

Jusque-là, nous pensons que la combinaison de la date et du numéro du permis de conduire sont suffisamment uniques pour identifier chaque transaction. Chaque transaction de location constituera un enregistrement de la « base de données location ». Il y aura un enregistrement de la base de données « Titre des films » pour chaque film loué. Ceci veut dire que si un client loue quatre films, ceux-ci viendront s'ajouter dans la base de données « Titre des films ». Ces titres sont « reliés » à l'enregistrement de location par le numéro de permis de conduire et la date.

Maintenant que nous avons vu cette astuce, nous réalisons cependant que l'employé, dans cet exemple, aura beaucoup de choses à taper. Toutes les informations du client et également tous les titres des films doivent être remplis. Heureusement, dans beaucoup de cas, nous pouvons diminuer la frappe en utilisant des informations déjà stockées dans l'ordinateur.

Beaucoup de clients qui louent des films seront membres du vidéo club de la boutique. La base de données des abonnés peut également être rattachée à la base de données location (lorsqu'un abonné loue des films) par son numéro d'abonné. Les données d'abonnement peuvent être copiées sur la base de données location, ce qui économise à l'employé à la fois du temps et des efforts (et diminue le risque d'erreur). On peut également avoir la chance d'ajouter des non-membres à cette base de données et, de ce fait, voir grossir la liste pour une future opération de mailing. Lorsqu'une rubrique sert à lier deux fichiers de base de données ensemble, la rubrique doit être de même taille et du même type dans les deux fichiers de base de données. Pour l'ordinateur, 'Alpha ' n'est pas la même chose qu'' Alpha' car celui-ci prend également en compte les espaces.

Les titres de films sont (ou devraient être) représentés par une rubrique dans la base de données d'inventaire de location de films. Les informations de base de l'inventaire comprennent :

Le titre du film
VHS ou BETAMAX
La position sur les rayons
La date d'achat ou de location par la boutique
Le prix d'achat ou tarif de location
Acquis ou loué
Le fournisseur

Le titre du film n'est pas pratique à utiliser pour manipuler les enregistrements des films. Un numéro d'identification permettra à l'employé de travailler plus facilement et de manière plus précise avec ou sans l'ordinateur. Dans notre exemple, nous admettrons qu'un numéro d'identification de cinq positions a été affecté à chaque film loué.

Si nous ajoutons quelques rubriques à la base de données d'inventaire, nous pouvons l'utiliser à la place de la « base de données des titres de films ».

Cotisation de location
Numéro de permis de conduire ou numéro d'abonné
Date de location
Nombre de passages en location
Loué (O/N)

Il existe un grand nombre de combinaisons possibles de fichiers pour cette application. Chacune a ses avantages et ses inconvénients suivant les particularités de l'application. Pour les besoins de l'exemple, nous utiliserons à partir de maintenant la base de données d'inventaire à la place de la base de données des titres de films.

Un diagramme de cette procédure de location ressemble à celle présentée par la Figure 15-9.

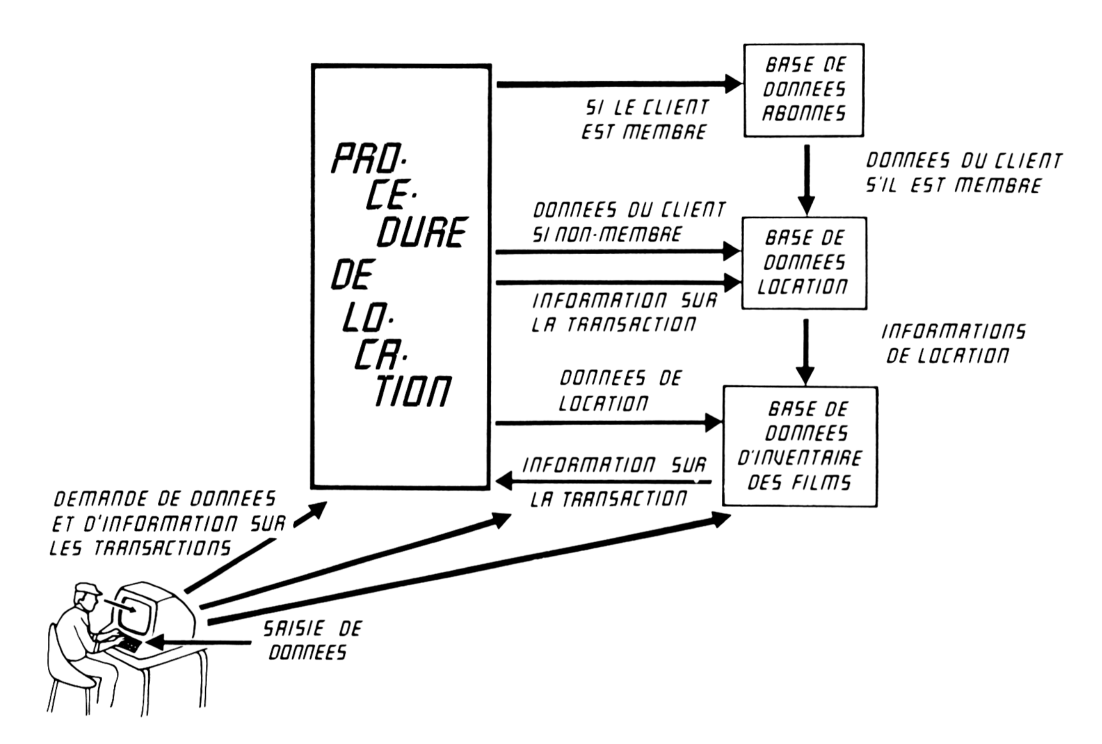


Figure 15-9

Voyons maintenant comment cela fonctionne. Lorsqu'un client loue un film, l'employé remplit un « formulaire informatique ». Ce formulaire devient un simple enregistrement dans la base de données location. La base de données inventaire de location sert à identifier les films qui sont loués. Le numéro de permis de conduire ou le numéro d'abonné et la date de location sont des rubriques communes aux fichiers de ces deux bases. Ces deux rubriques servent à lier ensemble les deux fichiers. A chaque location de film, la rubrique « nombre de passages en location » est incrémentée de 1. Si le client est un membre du club, la plupart des informations sont automatiquement retrouvées à partir de la base de données des abonnés.

A l'étape suivante, nous trouverons évidemment l'ordinateur en train de calculer le total de la cotisation de location et l'ajouter à la recette. Comme nous l'avons vu sur l'exemple précédent, le montant de la transaction peut être automatiquement ajouté au registre de caisse.

Si nous plaçons maintenant tous ces éléments ensemble, le diagramme de ce système peut se présenter comme sur la Figure 15-10.

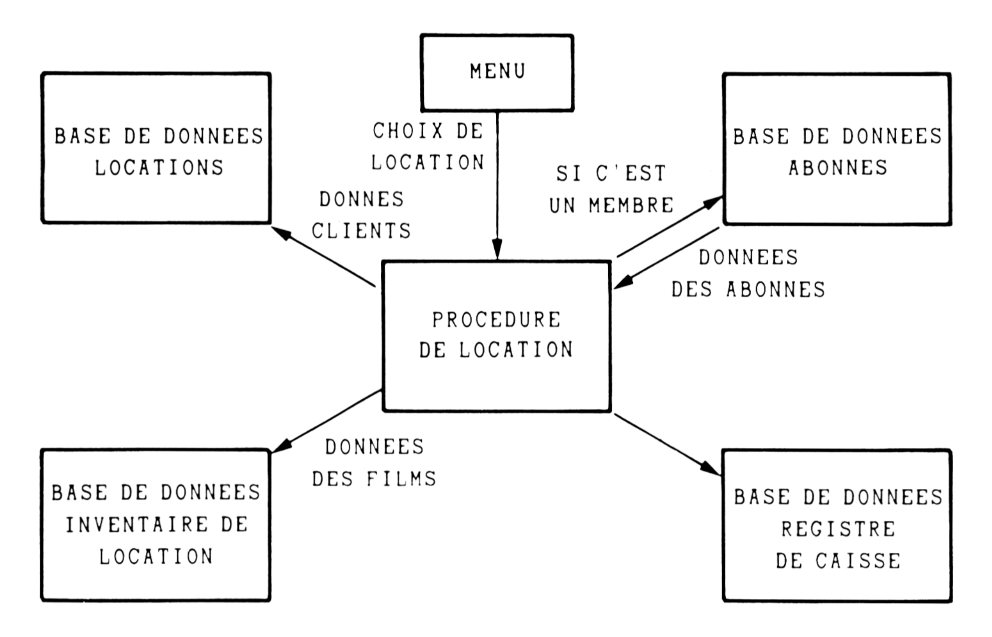


Figure 15-10

Le processus décrit ci-dessus est un exemple pratique montrant comment un système d'ordinateur peut servir à assister les opérations de gestion d'une petite affaire. Ce n'est pas le seul moyen, ni nécessairement le meilleur. Par contre, nous avons là une démonstration intéressante de la manière dont on peut résoudre un problème professionnel à l'aide d'un ordinateur.

L'étape suivante décrit la structure des fichiers de base de données utilisés dans ce processus. Le plan du fichier des abonnés a fait l'objet d'une description à la Figure 15-5. Les deux nouveaux fichiers de base de données sont présentés aux Figures 15-11 et 15-12. Une procédure qui exécutera le processus de location décrit ci-dessus est présentée à la Figure 15-3. Une mise en garde. Cette procédure sert à illustrer le processus. Elle a volontairement été simplifiée et ne pourrait pas servir telle quelle pour un besoin professionnel.

RUBRIQUE	DESCRIPTION	NOM DE RUBRIQUE	TYPE	TAILLE	DECIMALE(S)
1	Nom du client	CNOM	C	30	
2	Numéro et rue	CADRESSE	C	20	
3	Ville	CVILLE	C	20	
4	Code postal	CCPOST	C	5	
5	Numéro de téléphone	CTEL	C	8	
6	Identification	ID	C	8	
7	Cotisation de location	COTIS	N	5	2
8	Montant de la caution	CCAUTION	N	6	2
9	Nombre de locations	NBLOC	N	2	
10	Nombre de jours loué	NBJOURS	N	2	
11	Membre du club (Y/N)	CMEMBRE	L	1	
12	Date de location	CDATE	C	8	

Figure 15-11. Plan d'une base de données de location

RUBRIQUE	DESCRIPTION	NOM DE RUBRIQUE	TYPE	TAILLE	DECIMALE(S)
1	Titre du film	TITRE	C	30	
2	VHS ou BETA	VHS	L	1	
3	Location	RAYON	C	5	
4	Date d'acquisition	ACQDATE	C	8	
5	Cout	COUT	C	5	2
6	Acquis ou loué	ACQUIS	L	1	
7	Fournisseur	FOURNISS	C	20	
8	Identification	NO	C	8	
9	Montant de location	FRAISLOC	N	4	2
10	Date de location	LOCDATE	C	8	
11	Nombre de locations	NBLOC	N	3	
12	Loué (Y/N)	LOUE	L	1	
13	Numéro d'identification du film	NOFILM	C	5	

Figure 15-12. Base de données d'inventaire des films vidéo

La procédure décrite à la Figure 15-13 est la plus élaborée de ce livre. Bien qu'évoluée, elle ne présente pas de difficultés. La plupart des instructions servent à l'affichage écran.

Ces procédures introduisent une nouvelle commande — SELECT. Cette commande dBASE II permet à l'ordinateur de travailler avec deux fichiers de base de données en même temps. Jusqu'à maintenant, nous avions travaillé avec une seule base de données à la fois. Chaque fois que nous sélectionnons un fichier de base de données avec la commande USE, le système de base de données était positionné sur le premier enregistrement du fichier. Dans cet exemple, nous partirons de la base de données B :LOCATION vers d'autres fichiers de base de données et nous reviendrons à B :LOCATION. La commande SELECT nous permettra de « conserver notre position » dans le fichier B :LOCATION.

Lorsque nous utilisons deux fichiers de base de données simultanément, le premier est appelé le fichier PRIMAIRE, l'autre le SECONDAIRE. Nous pouvons nous connecter de l'un sur l'autre des deux fichiers sans perdre la place occupée précédemment dans l'autre fichier.

```
USE B:LOCATION
SET TALK OFF
APPEND BLANK
REPLACE CDATE WITH DATE()
ERASE
@1,15 SAY 'PROCEDURE DE LOCATION DE CASSETTES VIDEO'
@2,15 SAY 'LE CLIENT EST-IL UN MEMBRE DU CLUB (Y/N)' GET CMEMBRE
READ
CLEAR GETS
IF CMEMBRE
   @8,10 SAY 'TAPEZ LE NUMERO D'ABONNE' GET ID
   READ
   SELECT SECONDARY
   USE B:ABONNES
   LOCATE FOR NOMEMBRE=ID
   SELECT PRIMARY
   REPLACE CNOM WITH MEMBRENOM,CNORUE WITH NORUE, CVILLE WITH VILLE
   REPLACE CCPOST WITH CP,CTEL, WITH TEL
   ENDIF
```

```
  @3,10 SAY 'NOM DU CLIENT' GET CNOM
  @4,10 SAY 'NO ET RUE' GET CNORUE
  @5,10 SAY '    VILLE' GET CVILLE
  @6,10 SAY 'CODE POSTAl'GET CCPOST
  @7,10 SAY 'NO DE TELEPHONE' GET CTEL
IF CMEMBRE
  CLEAR GETS
ELSE
  @8,10 SAY 'NO DE PERMIS DE CONDUIRE' GET ID
  @9,10 SAY ' TAPEZ LE MONTANT DE LA CAUTION' GET CCAUTION
ENDIF

  @10,10 SAY 'NB DE JOURS DE LOCATION' GET NBJOURS
  @10,40 SAY 'NB DE FILMS LOUES' GET NBLOC
  READ
  CLEAR GETS
  SELECT SECONDARY
  USE B:VIDEOTEQ
  STORE NBJOURS TO X
  DO WHILE X>0
    STORE '       ' TO NF
    @12,10 SAY 'NO D'IDENTIFICATION DU FILM' GET NF
    READ
    LOCATE FOR NOFILM=NF
    REPLACE LOCDATE WITH DATE(),NBLOC WITH NBLOC+1,LOUE WITH Y
    REPLACE NO WITH ID
    SELECT PRIMARY
    REPLACE COTIS WITH COTIS+FRAISLOC
    STORE X-1 TO X
ENDDO
SELECT SECONDARY
USE B:REGISTRE
GO BOTTOM
IF.NOT.DATE=DATE()
  APPEND BLANK
  REPLACE DATE WITH DATE()
ENDIF
REPLACE CAUTIONS WITH CAUTIONS+CCAUTION
REPLACE LOCATION WITH LOCATION+COTIS
SELECT PRIMARY
```

```
ERASE
@5,10 SAY 'NOM DU CLIENT' GET CNOM
@7,10 SAY 'PRIX DE LA LOCATION' GET COTIS
IF CCAUTION>0
@7,40 SAY 'CAUTION DEMANDEE ?' GET CCAUTION
ENDIF
   STORE CCAUTION+COTIS TO DU
   @9,10 SAY 'MONTANT TOTAL DU' GET DU
   CLEAR GETS
   SELECT SECONDARY
   USE B:VIDEOTEQ
   DISPLAY OFF TITRE,FRAISLOC FOR NO=ID.AND.LOCDATE=DATE()
   SELECT PRIMARY
   SET TALK ON
   RETURN
```

Figure 15-13

Dans cet exemple, l'ordinateur contrôle d'abord pour voir si le client est un membre du club. Si c'est le cas, il récupère son numéro d'abonné au club, utilise le fichier des abonnés et saisit les données courantes telles que le nom, l'adresse, etc., dans le fichier Location. Si ce n'est pas le cas, il demande à l'employé de saisir chacune des données du client. Lorsque c'est terminé, il suggère à l'employé de saisir l'information concernant la transaction commerciale. A la suite de cela, il sélectionne le fichier vidéoteq et saisit les éléments de données nécessaires :

Date de location (LOCDATE)
Identifications du client (numéros de permis de conduire ou numéro d'abonné),

pour chaque film loué. De plus, il modifie la rubrique contenant le nombre de passages en location et indique que ce film est maintenant en location. Ensuite, il revient à la base de données Location et ajoute le montant de location du film à la rubrique Cotisation (COTIS). Une fois cette opération terminée, le « registre de caisse » est automatiquement mis à jour avec la cotisation de location et l'information concernant la caution.

Arrivé à l'étape finale, l'ordinateur affiche les éléments de base de la transaction comprenant le nom du client, le montant dû et les noms des films loués.

Cet exemple est particulièrement significatif. On utilise quatre fichiers de base de données séparés. La procédure vous évite (ainsi qu'à l'employé) toute manipulation de la base de données. Au fur et à mesure que l'on apporte de l'information, le système de gestion de base de données se déplace d'une base de données à une autre en ajoutant et en modifiant l'information suivant le cas.

La procédure B :LOCATION couvre la plupart des éléments constituant la gestion des locations d'une boutique vidéo. Vous ne devez pas la considérer comme opérationnelle pour gérer une affaire de location de films. La procédure ne prend pas en compte les multiples détails de cette activité. Certains éléments ont été laissés de côté pour permettre d'illustrer au mieux les concepts de travail avec des « formulaires écran » et sur plusieurs fichiers de base de données.

QUELQUES REMARQUES EN CONCLUSION

Tout au long de ce livre, nous avons pu démontrer que la gestion de base de données est un moyen aisé, efficace et naturel d'obtenir des résultats de votre ordinateur. En illustrant des concepts variés sur les bases de données, nous avons démontré par des exemples courants la souplesse des fonctions d'un système de gestion de base de données.

Avec du recul, nous réalisons que la « base de données » est un concept que nous connaissions. Le vocabulaire particulier lié aux systèmes de bases de données informatiques est peut-être moins connu mais n'est certainement pas compliqué. Il ne faut pas beaucoup de temps pour saisir qu'une colonne de données correspond à une rubrique. Ni beaucoup d'effort pour associer le titre de la base de données à son nom de fichier. Une fois que vous avez maîtrisé le vocabulaire de base, il vous est facile de maîtriser le système afin qu'il travaille pour vous.

C'est, évidemment, l'idée essentielle à retenir des systèmes de gestion de base de données micro-informatiques. Ils sont censés travailler pendant que vous réfléchissez. Il n'est pas nécessaire d'être un grand spécialiste des ordinateurs pour pouvoir faire travailler l'ordinateur à vos tâches routinières (vous avez vu dans cet ouvrage comme il est simple de faire travailler l'ordinateur). Que ce soit pour un usage professionnel ou personnel, un système de gestion de base de données fournit au débutant une excellente occasion pour, à la fois, optimiser son travail et retirer une certaine satisfaction dans l'usage d'outils logiciels efficaces.

Vous apprendrez certainement encore beaucoup de votre système de gestion de bases de données pour qu'il exécute des applications afin qu'il fournisse un meilleur rendement compte tenu de vos efforts. Rappelez-vous notre Stock de boutique d'alcools — nous avons certainement dépensé moins d'effort avec l'ordinateur, que s'il nous avait fallu faire l'inventaire d'une autre manière.

En commençant par des applications simples, vous observerez qu'il est facile de progresser vers des applications plus complexes — après tout, vous ne faites pas un marathon la première fois que vous courez sur piste.

Si vous faites l'acquisition d'un ordinateur et d'un système de gestion de base de données pour résoudre un problème complexe, une erreur courante consiste comme premier travail, à tenter imédiatement de le résoudre. Si c'est le cas, vous vous compliquez les choses (et la vie) inutilement. Il faut d'abord vous habituer à l'ordinateur et à votre logiciel.

Il vous sera difficile d'établir toute de suite la relation entre votre problème (particulièrement s'il est ardu), le matériel et le logiciel tout en même temps.

Pour vous familiariser avec votre système, nous vous suggérons de créer des problèmes « d'apprentissage » qui nécessitent des petites bases de données. Les exemples qui sont dans ce livre doivent pouvoir vous aider à construire des exemples personnels.

> LA SOLUTION AUX PROBLEMES COMPLIQUES SE COMPOSE DE PLUSIEURS ELEMENTS SIMPLES.

La construction par vous-mêmes d'exemples d'apprentissage utilisant des petites bases de données, vous permettra de contrôler facilement vos résultats pour vérifier si vous avez apporté la « bonne réponse ». On acquiert beaucoup d'expérience à travailler avec des bases de données de plus en plus grandes. Les exemples dans cet ouvrage sont suffisamment variés pour vous donner une bonne idée d'approche d'un problème particulier. Vous pouvez sélectionner les parties appropriées d'un ou plusieurs exemples et les assembler ensuite pour constituer la solution de votre problème.

Il est effectivement intéressant de travailler à partir d'exemples, mais vous apprendrez certainement plus rapidement en les utilisant comme un guide de travail dans la construction de vos applications. Après avoir sélectionné le problème à étudier, vous envisagerez sa construction, étudiez votre problème soigneusement et planifiez méthodiquement votre solution. Si vous comprenez votre problème et si vous le concevez méthodiquement, avec toutes les conséquences d'implantation sur votre système, vous réussirez toujours. Ces trois éléments de base — compréhension du problème, planification et implantation méthodique (pas à pas) — sont essentiels pour réussir avec les systèmes informatiques.

Réussir à résoudre un problème initialement « facile » vous donnera confiance et expérience pour vous attaquer à des problèmes de plus en plus complexes. Et si vous agissez de cette manière, vous découvrirez que ces problèmes complexes ne sont pas en définitive aussi difficiles. Adoptez une démarche par étape, habituez-vous au langage, et vous gagnerez.

Nous souhaitons que cet ouvrage soit un guide utile pour votre apprentissage à la maîtrise d'un outil aussi souple et utile que représente l'ordinateur. La gestion de base de données est probablement le moyen le plus sûr et le plus direct d'atteindre cet objectif. Avec elle, nous sommes sûrs que vous trouverez, dans l'ordinateur, un serviteur des plus précieux.

GLOSSAIRE

.AND.

Opérateur Booléen utilisé pour joindre deux expressions logiques de telle façon que l'expression qui en résulte s'applique aux deux caractéristiques de cette expression. Par exemple : NIVEAU='3'.AND.CLASSE='122' limite l'expression aux élèves de troisième affectés à la classe 122.

.NOT.

Opérateur Booléen utilisé pour signifier l'opposé de l'expression. Par exemple : .NOT.NIVEAU='3' signifie tous les niveaux sauf le troisième.

.OR.

Opérateur Booléen utilisé pour joindre deux groupes dans une expression logique. Par exemple, les niveaux troisième et quatrième peuvent être joints avec l'expression logique NIVEAU='3'.OR.NIVEAU='4'.

?

Commande dBASE II permettant d'afficher des éléments.

@

Opérateur dBASE II utilisé en conjonction avec les commandes SAY et GET pour générer un affichage sur l'écran ou sur l'imprimante.

*****

Utilisé habituellement pour indiquer la multiplication dans les systèmes d'ordinateur. Avec dBASE II, on l'utilise également pour indiquer que des enregistrements de la base ont été marqués pour effacement.

#

Le « signe dièse » a plusieurs fonctions. On l'utilise habituellement comme une abréviation de « non égal à ». Avec dBASE II, il est également utilisé comme symbole pour le « numéro d'enregistrement ». Par exemple : DISPLAY FOR # = 3 affichera les données du troisième enregistrement.

?

Symbole qui, dans dBASE II, indique à l'ordinateur de ne pas faire de différence entre les caractères minuscules et majuscules.

$

Opérateur dBASE II permettant de rechercher une séquence de caractères contenue dans une rubrique ou dans une variable mémoire. On l'appelle souvent une sous-chaîne ou un opérateur de chaîne. Elle peut être interprétée comme signifiant « contenu dans ». Par exemples, 'Robert'$NOM permet de rechercher la chaîne de caractères ROBERT contenue dans la rubrique NOM.

ACCEPT TO (nom de variable mémoire)

Commande dBASE II qui permet l'entrée de chaînes de caractères dans des variables mémoire désignées sans recours à des délimiteurs. S'utilise habituellement dans des procédures (fichiers de commande dBASE II).

ADL

Application Development Language — le langage de l'ordinateur utilisé avec dBASE II.

APPEND

Commande dBASE II pour ajouter des enregistrements à une base de données

APPEND BLANK

Variation de APPEND qui ajoute des enregistrements vides à une base de données. On l'utilise dans des procédures.

APPEND FROM <nom de fichier>

Utilisé pour ajouter des données à un fichier en cours d'utilisation à partir d'un fichier dont on a précisé le nom.

BACKUP

Processus de copie d'un disque ou d'un fichier disque vers un autre disque pour se protéger contre d'éventuelles pannes.

BASE DE DONNEES

Dépositaire d'un stock d'informations organisées de telle manière que celle-ci puisse facilement être retrouvée. Associée habituellement avec une base de données organisée stockée dans l'ordinateur utilisable par des applications multiples. Un exemple quotidien d'une base de données non informatique est l'annuaire du téléphone.

BASIC

Langage d'ordinateur. Abréviation de Beginners All Purpose Symbolic Instruction Code (Code d'instruction symbolique tous usages pour débutant).

BOOLEEN

Méthode de logique informatique basée sur le travail de Georges Boole qui a développé un certain type d'algèbre.

BOOTING

Processus de mise en route d'un ordinateur.

BROWSE

Commande dBASE II utilisée pour la modification. Plusieurs enregistrements peuvent être affichés simultanément pour une modification en mode plein écran.

BYTE

Capacité de mémoire requise pour stocker un caractère comme un « A », « # » ou « 9 ».

CANCEL

Commande utilisée pour terminer une procédure et reprendre le contrôle de l'ordinateur à partir du clavier.

CASE

Variation de l'instruction IF. Case s'utilise uniquement avec DO CASE/ENDCASE.

CHANGE

Commande dBASE II utilisée pour la modification.

CARACTERE

Chaque symbole du clavier peut être imprimé comme un « A », un « $ », un « 1 » ou un « a » (et inclure des espaces blancs).

RUBRIQUE CARACTERE

Une rubrique de la base de données qui va servir à contenir des caractères.

CHAINE DE CARACTERES

Une séquence continue de caractères telles que « Johnny Guitare ».

CHR()

Commande dBASE II qui vous permet de contrôler directement des périphériques tels que l'imprimante et l'écran.

CLEAR

Cette commande remet tout à zéro. Toutes les bases de données en utilisation (USE) sont refermées, les variables mémoires supprimées, le système se présente alors comme si vous veniez d'entrer dans dBASE II.

CLEAR GETS

Commande qui supprime tous les GET en attente sans altérer l'écran (comme le ferait la commande ERASE). Ceci permet de limiter l'étendue de l'instruction READ pour utiliser uniquement les GET activés après la commande CLEAR GETS.

COBOL

Langage d'ordinateur utilisé de façon intensive dans des applications professionnelles sur gros et moyen systèmes. Le premier langage d'ordinateur se rapprochant de l'anglais. Abréviation de Committee On Business Oriented Languages (Langage orienté business).

COMMANDE

Vocabulaire dBASE II d'une instruction d'ordinateur.

FICHIER DE COMMANDE

Jeu d'instructions d'ordinateur rangé ou sauvegardé sur un disque pour un usage répétitif. Terminologie dBASE II pour une procédure informatique.

CONDITION

Expression logique qui peut être utilisée pour définir explicitement une commande telle que DISPLAY. DISPLAY FOR NOM = 'JOHNNY GUITARE' modifie la commande DISPLAY de telle façon qu'elle s'applique uniquement aux enregistrements respectant la condition NOM = 'JOHNNY GUITARE'.

REMPLACEMENT CONDITIONNEL

Technique de modification du contenu d'une base de données où la commande REPLACE est modifiée par une condition. REPLACE PROFESSEUR WITH 'MARTIN' FOR CLASSE = '171' remplacera le contenu de la rubrique PROFESSEUR par 'MARTIN' dans tous les enregistrements remplissant la condition CLASSE = '171'.

CONTINUE

Commande se positionnant sur le prochain enregistrement qui satisfait à une condition spécifiée dans la commande LOCATE.

TOUCHE DE CONTROLE

Touche qui fournit potentiellement une troisième valeur à toutes les touches du clavier. Similaire à la touche Shift qui fournit une seconde signification à toutes les touches du clavier.

COPY TO <nom de fichier>

Commande dBASE II qui copie la base de données en cours dans la base de données précisée par le nom de fichier. Utilisée pour créer une copie d'une base de données de sauvegarde. Il existe des variations qui permettent de copier uniquement certaines parties de la base de données en cours dans une base précisée par le nom de fichier.

COUNT

Commande d'affichage de données qui compte le nombre d'enregistrements qui satisfont à une expression conditionnelle.

CP/M

Contrôle Programme pour micro-processeurs. Système d'exploitation populaire pour micro-ordinateur. CP/M est une marque déposée de Digital Research Corporation.

CPU

Central Processing Unit (Unité centrale de traitement) — le processeur central du système d'ordinateur. Il contient une unité arithmétique, des registres spéciaux et une unité principale de stockage.

CREATE

Commande dBASE II permettant d'établir la structure d'un nouveau fichier de base de données.

CRT

Cathode Ray Tube (Tube à rayon cathodique). Utilisé couramment pour signifier un périphérique d'affichage vidéo utilisé en liaison avec des ordinateurs.

CTRL KEY

Touche de contrôle — Utilisée en combinaison avec diverses touches pour leur fournir une troisième fonction.

DASD

Direct Access Storage Device — Périphérique de stockage à accès direct tel qu'un disque.

DATA

Partie d'information. S'utilise très rarement comme élément isolé. Peut constituer de l'information lorsqu'elle est utilisée avec d'autres éléments de données. Par exemple le numéro de téléphone 555-3213 n'a pas d'utilité par lui-même. Il communique de l'information lorsqu'il est utilisé en conjonction avec un autre élément de données, par exemple, Martin Johnson.

BASE DE DONNEES

Dépositaire d'un stock d'informations organisées de telle manière que celle-ci puisse facilement ête retrouvée. Associée habituellement avec une base de données organisée stockée dans l'ordinateur utilisable par des applications multiples. Un exemple quotidien d'une base de données non informatique est l'annuaire du téléphone.

SGBD

Système de gestion de base de données.

DEFAUT

Lorsque l'ordinateur reçoit une commande, il doit entreprendre une action. Sauf indication contraire, il effectue une action préprogrammée. Cette dernière est appelée une « action par défaut ».

DELETE

Commande dBASE II qui marque les enregistrements pour effacement (voir également PACK).

DELIMITEURS

Moyen d'identification de chaîne de caractères dans l'ordinateur. Par exemple, elle permet à l'ordinateur de distinguer la rubrique NOM du mot « NOM ».

LECTEUR DE DISQUE

Périphérique mécanique utilisé pour lire et écrire de l'information sur un disque.

DISQUES

Plateau circulaire, recouvert d'oxyde magnétique pour stocker les données. Support sur lequel un ordinateur, de façon permanente ou temporaire, range des informations qui seront utilisées ou relues à un moment ultérieur.

DISPLAY

Commande dBASE II permettant d'afficher le contenu d'un enregistrement de données.

DISPLAY ALL

Variation de la commande DISPLAY. Cette option affiche tous les enregistrements de la base de données en cours — s'arrête tous les 15 enregistrements.

DISPLAY FILES ON <lettre d'identification du lecteur>

Commande dBASE II qui affiche le nom de fichier, le nombre d'enregistrements et la date de la dernière modification de tous les fichiers de base de données sur le disque spécifié.

DISPLAY FOR <condition>

Variation de la commande DISPLAY. La clause FOR <condition> modifie la commande DISPLAY de telle façon que tous les enregistrements respectant la condition soient affichés par groupe de 15.

DISPLAY MEMORY

Option de la commande DISPLAY qui affiche le nom, le type, la taille et le contenu de toutes les variables mémoire.

DISPLAY OFF

Option de DISPLAY qui affiche l'enregistrement de la base de données sans le numéro d'enregistrement.

DISPLAY STRUCTURE

Option de la commande DISPLAY utilisée pour afficher la structure d'une base de données en cours.

DO <nom de fichier>

Commande dBASE II indiquant à l'ordinateur d'exécuter une procédure précisée par son nom (fichier de commande).

DO CASE

Commande dBASE utilisée comme alternative aux instructions IF, ENDIF imbriquées. Elle doit être accompagnée d'une commande ENDCASE. Utilisée habituellement dans les procédures.

DO WHILE <condition>

Commande dBASE II indiquant à l'ordinateur de répéter l'exécution d'une séquence de commandes entre une ligne DO WHILE et sa conclusion ENDDO aussi longtemps qu'une condition est remplie. Utilisée dans les fichiers de commande dBASE II (procédures).

DOS

Système d'exploitation disque (Disk Operating System) — logiciel.

EDIT <numéro d'enregistrement>

Commande permettant de modifier le contenu d'une rubrique de données. Dans dBASE II, elle invoque l'opération en mode plein écran qui vous permet de modifier le contenu des rubriques de votre choix en déplaçant le curseur sur l'emplacement approprié et en tapant la nouvelle donnée.

EJECT

Commande dBASE II qui envoie une page sur l'imprimante.

ELSE

Commande utilisée dans une procédure. Fournit une alternative à l'exécution d'une commande à l'intérieur d'une structure IF.

ENDCASE

Commande de procédure qui termine un DO CASE.

ENDDO

Commande de procédure qui termine un DO WHILE.

ENDIF

Commande de procédure qui termine une commande IF.

EOF

Fin de fichier (End of file) — fonction dBASE II ; employé également comme caractère spécial dans un fichier ASCII.

ERASE

Commande qui efface l'écran vidéo.

ESCAPE

Touche utilisée sur beaucoup d'ordinateurs vous permettant d'interrompre une procédure ou l'exécution d'une commande à partir du clavier. Sur les ordinateurs qui n'ont pas la touche ESCAPE, on utilise la touche CTRL.

RUBRIQUE

Dans les systèmes tels que dBASE II, une rubrique contient un élément d'information. Elle correspond à une colonne d'informations sur une base de données papier. Son synonyme en vocabulaire informatique est le mot CHAMP.

DESCRIPTION DE LA RUBRIQUE

La description d'une rubrique comprend trois parties : le nom, le type et la taille de la rubrique incluant le nombre de positions décimales (s'il y en a).

NOM DE RUBRIQUE

Le titre de la rubrique. Contient de un à dix caractères, commence par une lettre, et ne doit pas contenir d'espaces blancs.

TAILLE DE RUBRIQUE

Le nombre de positions caractères nécessaires pour contenir la donnée qui sera placée dans la rubrique.

TYPE DE RUBRIQUE

Le type de données qui seront stockées dans les rubriques. Il y a trois types de rubrique possibles : caractères, numériques et logiques.

LARGEUR DE RUBRIQUE

Le nombre d'espaces nécessaires à la rubrique pour contenir les données.

FICHIER

Collection d'informations telle qu'un fichier de données ou un fichier de commande, stockée sur une unité de disque identifiable.

NOM DE FICHIER

L'ordinateur en a besoin pour identifier le fichier. Doit être constitué de un à huit caractères au maximum, commencer par une lettre et ne pas contenir des espaces.

TYPE DE FICHIER

Extension de trois caractères au nom de fichier, précédée par un point. Sert à distinguer les différents types de fichiers possédant le même nom. L'ordinateur peut alors reconnaître le type de fichier avec lequel il doit travailler d'une certaine manière. Par exemple, un fichier .DBF est reconnu par dBASE II comme un fichier de base de données.

FIND <clé>

Commande dBASE II servant à retrouver rapidement un enregistrement à partir d'une clé. S'utilise exclusivement avec des fichiers indexés.

FLOPPIES

Ce sont des disques souples (disquettes). Utilisés couramment avec les micro-ordinateurs . L'information est stockée sur une mince feuille de mylar souple.

FORTRAN

Langage classique d'ordinateur principalement utilisé dans les applications scientifiques. Abréviation de Formula Translation (Langage traducteur de formule).

GET

Commande dBASE II utilisée avec la commande @ permettant d'afficher le contenu d'une rubrique ou d'une variable mémoire. Si on l'emploie avec la commande READ, on modifie le contenu de la variable mémoire simplement en tapant la nouvelle information.

GO BOTTOM

Commande qui permet à dBASE II de se positionner sur le dernier enregistrement de la base de données en cours.

GO TOP

Commande qui permet à dBASE II de se positionner sur le premier enregistrement de la base de données en cours.

GOTO <numéro d'enregistrement>

Cette commande sert à repositionner le pointeur d'enregistrement de la base sur le numéro spécifié dans la commande.

DISQUE DUR

Disque constitué d'un plateau rigide en métal recouvert d'oxyde magnétique ; peut stocker une grande quantité de données.

E/S (I/O)

Dispositif par lequel l'ordinateur accepte des informations des différentes unités périphériques. Abréviation d'Entrée-Sortie.

IF <condition>

Instruction indiquant à l'ordinateur d'exécuter un ensemble de commandes uniquement dans le cas où la condition est remplie.

INDEX ON <liste des rubriques> TO <nom de fichier>

Commande dBASE II permettant de créer un fichier index nommé par le nom de fichier. Le fichier index fait apparaître le contenu des rubriques indiquées dans l'ordre alphabétique.

INPUT TO <variable mémoire>

Commande dBASE II permettant la saisie au clavier d'informations numériques dans des variables mémoire. Seules les données numériques seront acceptées. Utilisée couramment dans les procédures.

INSERT

Commande qui donne la possibilité d'insérer un nouvel enregistrement de données au milieu de la base de données.

INT ()

Fonction qui arrondit le nombre décimal placé entre parenthèses et supprime tout ce qui est à droite du point. Abréviation de integer (entier).

JOIN

Commande qui assemble deux bases de données pour en créer une troisième selon certains critères.

CLAVIER

Périphérique ressemblant au clavier d'une machine à écrire et permettant par son utilisation de « dialoguer » avec l'ordinateur.

LEN (nom de la variable mémoire)

Fonction dBASE II qui indique la longueur en nombre de caractères d'une chaîne de caractères placée dans la variable mémoire.

LIST

Commande dBASE II permettant d'afficher tous les enregistrements d'une base de façon continue.

LOCATE FOR <condition>

Commande dBASE II utilisée pour retrouver l'enregistrement qui satisfait la condition.

RUBRIQUES LOGIQUES

Rubriques qui ne peuvent avoir que deux contenus possibles. Utilisée pour saisir des données mutuellement exclusives telles que « T » et « F » ou « Y » et « N », ou « M » et « F » (Vrai/Faux, Oui/Non).

ENREGISTREMENTS LOGIQUES

Ce sont des enregistrements dans un fichier index ou dans un fichier similaire qui représentent l'image d'une partie des données en cours. Utilisés habituellement comme aide dans l'utilisation des données.

LOOP

Commande qui provoque le retour au début d'un DO WHILE dans l'exécution d'un fichier de commande. Sert d'échappement lorsqu'une condition indésirable est rencontrée.

MEMOIRE DE MASSE

Dispositif de stockage périphérique tel qu'un lecteur de disque ou un lecteur de bande magnétique.

VARIABLE MEMOIRE

Vous permet de stocker l'information dans l'ordinateur en mémoire centrale pour un usage temporaire. C'est un peu la même chose que la mémoire d'une calculette. Toutes les informations sont perdues lorsque l'on éteint la machine.

MENU

Procédure informatique qui affiche un jeu de choix en vue d'une action.

SYSTEME DE MENU

Procédure informatique qui permet à l'ordinateur d'exécuter certaines actions au choix de l'utilisateur à partir d'une sélection dans un menu.

MICRO-ORDINATEUR

Petit ordinateur construit principalement pour un usage personnel.

MILLISECONDE

Un millième de seconde.

MINI-ORDINATEUR

Un ordinateur généralement configuré pour un emploi simultané par un petit nombre de personnes. Nettement plus grand et plus puissant qu'un micro-ordinateur.

MODIFY STRUCTURE

Commande dBASE II qui vous permet de modifier la structure d'une base de données. En modifiant la structure, vous détruisez généralement le contenu de la base de données. Cette commande doit être utilisée avec précaution.

MODIFY COMMAND

Commande dBASE II qui vous permet de créer et/ou modifier le contenu d'une procédure (fichier de commande).

NOTE <texte>

Commande dBASE II (généralement utilisée dans les procédures). Lorsque l'ordinateur rencontre cette commande, il ignore le texte qui suit. Utilisée pour décrire la procédure à des fins de documentation.

RUBRIQUES NUMERIQUES

Rubriques qui contiennent des nombres significatifs pour être utilisés dans des calculs numériques.

SYSTEME D'EXPLOITATION

Logiciel permettant de gérer le système matériel de votre ordinateur.

PACK

Commande qui EFFACE physiquement tous les enregistrements marqués pour effacement.

PASCAL

Langage d'ordinateur possédant certaines fonctions de dBASE II.

PERIPHERIQUES

Dispositifs utilisés avec l'ordinateur tels qu'une imprimante et des lecteurs de disque.

ENREGISTREMENTS PHYSIQUES

Les enregistrements de données en cours.

PIP

Commande CP/M qui vous permet de copier des fichiers informatiques d'un disque à un autre.

PL/1

Langage d'ordinateur.

POINTEUR

Mécanisme logiciel qui dirige l'ordinateur sur l'enregistrement de la base de données qui nous intéresse.

CLE PRIMAIRE

L'unique moyen d'identifier l'enregistrement directement utilisable par le système de l'ordinateur. Avec dBASE II, la CLE PRIMAIRE est le numéro d'enregistrement.

IMPRIMANTE

Dispositif périphérique qui sort les données de l'ordinateur sur papier.

PROGRAMME

Série de commandes qui seront exécutées par l'ordinateur comme un tout. Une procédure est un programme. Dans dBASE II, un fichier de commande est un programme.

PROGRAMMEUR

Toute personne qui écrit des programmes.

PROMPT (SUGGESTION)

Indication fournie par l'ordinateur qui est prêt à accepter des commandes au clavier. C'est également une requête pour suggérer à l'utilisateur d'entrer des informations particulières à partir du clavier.

QUERY (INTERROGATION)

Demande d'information formulée par l'utilisateur à l'ordinateur. Il s'agit généralement d'une demande frappée au clavier.

QUIT

Commande qui provoque la fermeture de tous les fichiers en utilisation et la sortie vers le système d'exploitation.

RAM

Abréviation de Random Access Memory. Il s'agit de la mémoire centrale de l'ordinateur. La mémoire à accès aléatoire est vraiment la mémoire de l'ordinateur directement adressable et sur laquelle on peut lire ou écrire.

READ

Commande dBASE II utilisée en même temps que la commande GET permettant de mettre en place les possibilités d'édition en mode plein écran des variables mémoire et des rubriques de données.

RECALL

Commande dBASE II qui « rappelle » les enregistrements précédemment marqués pour effacement.

ENREGISTREMENT

Unité fondamentale d'éléments de données. Dans dBASE II, c'est cette information qui est contenue sur une ligne dans un tableau rectangulaire de lignes et de colonnes.

VERROUILLAGE D'ENREGISTREMENT

Artifice servant à empêcher les modifications simultanées d'un simple élément de données à partir de systèmes pouvant regrouper plusieurs utilisateurs en même temps.

NUMERO D'ENREGISTREMENT

Numéro d'identification affecté à chaque enregistrement de données par le système de gestion de la base. Le numéro d'enregistrement est unique pour chaque enregistrement (deux enregistrements ne peuvent avoir le même numéro).

SYSTEME DE BASE DE DONNEES RELATIONNEL

Un type de système de gestion de base de données basé sur l'utilisation d'un tableau rectangulaire de lignes et de colonnes. Différents fichiers de base de données peuvent être reliés par le contenu d'une rubrique de données, les contenus de rubriques peuvent être communs aux deux fichiers de base de données.

RELEASE <noms de variables mémoire>

Commande dBASE II qui « efface » les variables mémoire indiquées.

RENAME <nom de fichier 1> TO <nom de fichier 2>

Commande dBASE II permettant de renommer un fichier. Ce fichier ne doit pas être en utilisation.

REPLACE <nom de fichier> WITH <nouveau contenu>

Commande dBASE II qui remplace le contenu d'une rubrique de données spécifiée, par un nouveau contenu. Utilisée la plupart du temps dans des procédures. Peut tout de même s'utiliser à partir du clavier en conjointement avec FOR <condition>.

REPORT

Commande dBASE II qui prépare des états d'impression basés sur le contenu d'une base de données et à partir de réponses à une série de questions simples. Le document imprimé est affiché sur l'écran. Les réponses aux questions sont saisies à partir du clavier lorsque l'on utilise un document spécifique pour la première fois. Les réponses sont sauvegardées dans un fichier .FRM et le document imprimé sera par la suite généré automatiquement. Il peut y avoir plusieurs documents imprimés disponibles à tout moment.

REPORT TO PRINT

Variation de REPORT qui provoque l'impression du document.

RETURN

Commande qui termine un fichier de commande et retourne à la procédure appelante. « Redonne la main » au clavier s'il n'y a pas d'autre procédure.

TOUCHE RETURN

Similaire à la touche de retour du chariot d'une machine à écrire. Est également appelée ENTER sur les micro-ordinateurs AMSTRAD.

ROM

Abréviation de Read Only Memory (Mémoire à simple lecture).

RUB

Touche qui efface les caractères à gauche du curseur.

SAVE

Commande qui copie le contenu défini des variables mémoire vers une mémoire de masse.

SAY

Commande dBASE II utilisée en conjonction avec l'opérateur @ pour afficher une information sur l'écran ou sur l'imprimante.

SELECT PRIMARY

Commande qui sélectionne la base de données PRIMAIRE.

SELECT SECONDARY

Commande qui sélectionne la base de données SECONDAIRE.

ACCES SEQUENTIEL

Méthode d'accès aux enregistrements de données en séquences par l'ordinateur — commençant au premier enregistrement — jusqu'à ce que soit trouvé l'enregistrement souhaité.

SET ECHO ON/OFF

Toutes les commandes provenant d'un fichier de commande sont reproduites en écho sur l'écran comme si elles venaient d'être entrées à partir du clavier. ECHO est normalement éteint et doit être allumé si nécessaire.

SET PRINT ON/OFF

Dirige les sorties de l'ordinateur vers l'imprimante. Habituellement les sorties ne sont pas dirigées vers l'imprimante.

SET TALK ON/OFF

Les résultats de commandes sont habituellement affichés sur l'écran. C'est quelquefois indésirable lorsque l'on utilise des fichiers de commande.

SKIP n

Commande qui positionne la base de données en avant ou en arrière d'un nombre n d'enregistrements.

SORT

Commande qui provoque la réorganisation physique de la base de données dans un ordre particulier — généralement dans l'ordre des valeurs numériques d'une rubrique ou dans l'ordre alphabétique sur le contenu d'une rubrique caractère.

STORE

Commande qui stocke les données dans les variables mémoire.

STRUCTURE

Organisation prédéfinie de votre base de données. Elle est établie à partir du nom, du type et de la taille de la rubrique.

SUM <nom de rubrique>

Commande qui ajoute les contenus des rubriques indiquées de tous les enregistrements.

TERMINAL

Le moyen par lequel vous communiquez avec votre ordinateur et avec lequel celui-ci communique avec vous. Le dispositif terminal le plus populaire en micro-informatique est le terminal à affichage vidéo.

TYPE

Cette fonction ramène le type de la donnée, c'est-à-dire « C », « N » ou « L » suivant qu'il s'agit respectivement de Caractère, Numérique, Logique, ou « U » indéfini.

UPDATE

Commande qui regroupe ensemble les enregistrements de deux bases de données.

USE <nom de fichier>

Commande qui indique à dBASE II avec quelle base de données vous voulez travailler.

VAL(x)

Permet au contenu d'une rubrique de données du type caractère ou d'une variable mémoire d'être utilisé dans des calculs arithmétiques.

ECRAN VIDEO

Dispositif d'affichage vidéo associé à votre ordinateur.

WAIT

Commande d'interruption dans un fichier de commande permettant de stopper le processus de traitement et d'attendre l'entrée d'un caractère unique à partir du clavier avec un message indiquant l'attente (WAITING).

WAIT TO <nom de la variable mémoire>

Similaire à WAIT excepté la clause TO qui permet au caractère saisi au clavier d'être stocké dans la variable mémoire indiquée.

Imprimé en France
Décembre 1985
DM IMPRESSIONS
1, place de l'Église
BP 38 92215 Saint-Cloud
(1) 47 71 25 90

LE LIVRE

Initiation aux bases de données pour micro-ordinateurs - Application à dBASE II pour Amstrad CPC 6128 et PCW 8256 décrit le logiciel d'ASHTON-TATE qui est le système de gestion de base de données, le plus répandu au monde. Ce livre explique comment créer des fichiers, manipuler les données et employer les commandes. Le lecteur apprendra, en langage clair, comment préparer, créer, modifier, améliorer et utiliser une base de données, ainsi que la manière de mettre en œuvre dBASE II pour des applications de gestion. De nombreuses images d'écran conçues par dBASE II illustrent cet ouvrage. dBASE II est également un puissant langage de programmation.

L'AUTEUR

Il est l'auteur le plus connu d'ouvrages sur dBASE II.
Robert A. BYERS a été Directeur du Département de Contrôle au Laboratoire de propulsion à réaction de PASADENA, Californie. Il est une autorité incontestée du langage de programmation dBASE.

La Commande Electronique
7, RUE DES PRIAS — 27920 SAINT-PIERRE DE BAILLEUL
TÉL. 32 52 54 02 TÉLEX LCE 180 855

Prix 250 F TTC

ISBN 2-905350-06-7

INITIATION AUX BASES DE DONNEES

application à dBASE II pour AMSTRAD

Robert A. Byers

ASHTON·TATE



Document numérisé avec amour par :

AMSTRAD CPC

MÉMOIRE ÉCRITE

https://acpc.me/